CREW SCHEDULING METHODS
IN OPERATIONAL PLANNING OF TRANSPORTATION SYSTEMS

by

Jean-Marc Rousseau

Presented at the "1st Course on Transportation Planning Models of the International School of Transportation Planning (I.C.T.S.)", Amalfi, Italy, October 11-16, 1982.

Université de Montréal

Centre de recherche sur les transports - Publication #264
Département d'informatique et de
recherche opérationnelle - Publication #450

Août 1982

Dépôt légal - 3e trimestre 1983

THE PROBLEM

Manpower scheduling problems in transportation systems arise in two main contexts: in the scheduling of vehicle operators and in-vehicle crews, and in the scheduling of manpower to service vehicles, passengers or freight at terminals (airports, intercity bus terminals, bus stations, subway stations, freight terminals and transfer points, etc.).

SCHEDULING OF VEHICLE OPERATORS AND IN-VEHICLE CREWS-The major constraint for this problem is the vehicle schedule which must be respected. In airline, train or intercity bus crew scheduling, it may happen that a crew spends one or several nights (or rest periods) outside its home city (crew base). A *duty period* generally comprises all work assignments during a workday between two rest periods. The set of duties between two rest periods at the crew base is referred to as a *pairing* or *rotation*. The union contract rules distinguish between duty time (time between the beginning and the end of a workday) and in-vehicle time. These times are generally paid differently or their ratio must be constrained.

For example, in air crew scheduling, pilots require that their credited time (time for which they are paid) for a duty period be equal to the maximum of a set minimum, the actual in-flight time of duty and half the duty time. In urban transit, an excess of duty time over in-vehicle time of more than two hours is generally paid for.

When a crew changes vehicle, it may be deadheaded to a different location to take on a new assignment. In airlines, the crew will travel on a flight as passengers. In urban transit, crews either walk or take the bus. In airlines, deadheading is done only when absolutely necessary because it is costly. In urban transit, it is quite a general practice because of the smaller distances involved. However, some cities (Paris, for example) do not permit *interlining* (the possibility for a driver of driving vehicles on several different buses lines).

While union contract rules in the air industry tend to be similar from one country to another, they vary considerably from city to city in the urban transit industry

In addition to the construction of duty periods and pairing, rosters (or blocks in airline jargon) must be produced. These consist of weekly or monthly (or any other period) assignments of duties or pairings together with the rest days for crew members. In some cases (fixed location scheduling) this consists simply of assigning two rest days per week, the duties performed every workday being similar or at least of equal quality. On most North-American transit companies the choice of a roster is made by the drivers themselves by order of seniority. The most junior drivers take whatever is left in terms of rest days and duties. In European urban transit, rosters are constructed so that each driver has a set of duties and rest periods of equal quality over a given period. As all drivers are paid the same amount, they all have the same quality of life. In airlines, the pilot association is often responsible for the production of rosters from the pairings produced by the company. Some of these rosters are often of very high quality for the most senior pilots, low quality rosters being left for more junior pilots.

Transit Crew Scheduling-The difficulty of the transit crew scheduling problem arises directly from the service that a transit company must offer and the travel patterns of the population. Fig.1 illustrates different patterns of service level by time of day for Québec, Montréal (weekend, one garage), Florence (one garage) and Ravenna (weekend).

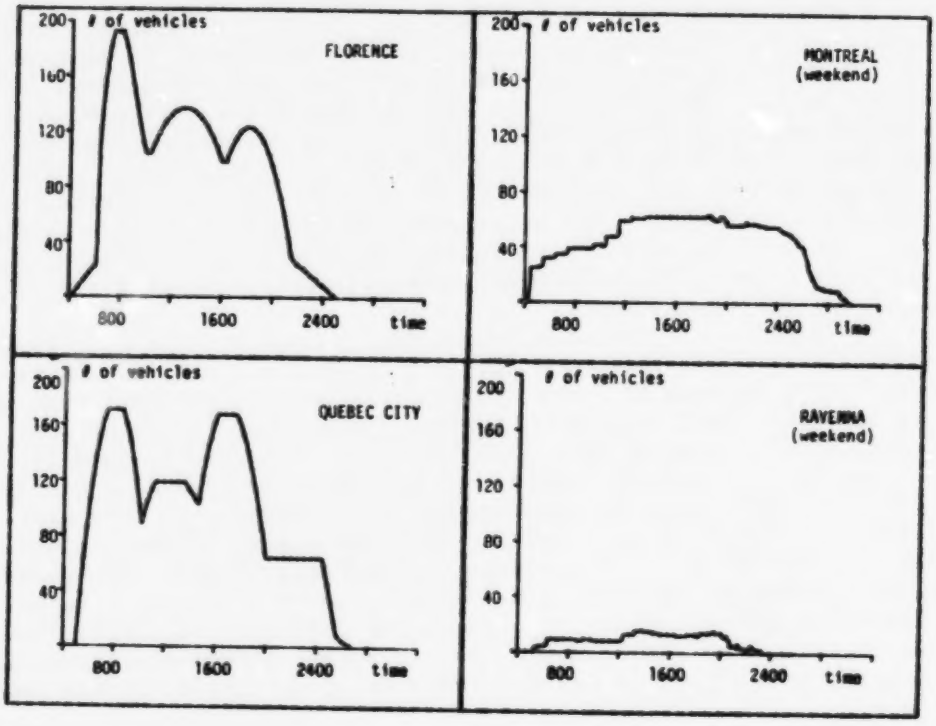


Figure 1 - Service level by time of day [15]

Note that the number of vehicles in service may be twice (even three times is some cases) as great at peak hours as at off-peak hours. This obviously necessitates either part-time drivers, or split-shift workdays for full-time drivers, or both. In most companies, unions are refusing or severely restricting the use of part-time drivers. Several rules have also appeared defining legal split-shifts and working conditions, and limiting the number of less desirable workdays and/or compensating the drivers for them. These working conditions are described in more detail in several papers [44,51].

The rules listed in Table 1 illustrate the complexity of the problem of constructing a minimal cost schedule.

Table 1 - Examples of Drivers' Union Contract Rules in Transit Companies [11]

	Definition	Examples of rule
Relief point	: is a point where a change of drivers can be made on a bus route	Most of the time there is only one predetermined point on each bus route
Guaranteed minimum	: minimum number of hours paid per working day	8 hours
Piece of work	: refers to the period of time during which a driver works with the same vehicle	- not more than three pieces of work in a day - minimum of two hours paid for a piece of work - there is a maximum percentage of working days with a piece over six hours
Break	: free period between two pieces of work	- at most one unpaid break - break must be at least ½ hour if there is a break
Spread	: the elapsed time between the start of the first piece of work to the end of the last piece of work in a working day	- maximum spread 12 hours - maximum spread for working days ending after 23h00 is 10 hours - spread over 10 hours is compensated
Straight run	: an assignment made out of only one piece of work or in several companies an assignment with a small spread (8h30-9h00)	a minimum or a maximum percentage of working days varying from 20% to 60% must be straight runs
Travel time	: time spent to travel between pieces of work or between relief point and the garage	- some times paid totally or partly - predetermined time must be allowed

Airline Crew Scheduling-The difficulty of air crew scheduling arises mostly from the aircraft schedule itself which may force long duty periods or long trips away from the crew operating base together with unproductive deadheading. The union contract includes several rules concerned with security and costs to heavily penalize unpleasant working conditions.

The constraints of the air crew scheduling problem are a combination of federal regulations, union rules and company policies. Some constraints are very similar to the mass transit crew scheduling problem just discussed. A *duty period* is a continuous block of time when the crew is on duty. The basic regulations governing a duty period in the U.S. are as follows. A duty period begins one hour before a flight and ends half an hour after the flight. A duty period can be non longer than 12 hours and a crew can fly a maximum of eight hours within the duty period. A crew may wait between flights in a single duty period as long as that wait does not exceed three hours. If a wait exceeds three hours, a crew rest is declared, the duty period is ended and this crew rest must be at least 9½ hours in duration. Federal regulations also prohibit any crew from flying more than eight hours in any 24 hour period (called the 8 in 24 constraint). There may also be 16 in 48 and 24 in 72 constraints.

A duty period is made up of a set of *flight legs*. A flight leg corresponds to a flight between two cities, departing one city at a specified time and arriving at the second city at a specified time. Flight legs make up the set of input tasks for this scheduling problem. Constraints on the formation of duty periods limit the number of flight legs in a duty period to seven and require at least one hour for a crew to transit a base and change aircraft.

Vehicle Versus Crew Schedule-Although we have described crew schedules in terms of vehicles, this by no means suggests that vehicles should be scheduled prior to and separate from crews. Unfortunately, in nearly all practical settings this is the approach taken. In the case of airline and air crew scheduling, this approach makes sense since airplane costs significantly dominate air crew costs. In the mass transit case, however, crew costs dominate vehicle operating costs and in some cases reach as high as 80% of total operating costs. Under such circumstances it seems foolish to allow vehicles to be scheduled independently of and without regard to crews.

We present one approach which tries to combine these two problems; in most of this paper however, the vehicle schedule will be considered fixed.

Although the primary thrust of this paper is toward scheduling crews to operate vehicles, we start with a simpler case, that of scheduling workers at a fixed location. Considering this case will help in the conceptualization of more complicated situations.

SCHEDULING TERMINAL MANPOWER-This problem is generally simpler because all scheduling is done in a same given location. It is assumed that workers are interchangeable, that any worker can be relieved at the end of any time period and that any worker can start at the beginning of any time period. The problem arises because of peak demands for vehicles arising from the vehicle schedule which has been tailored as much as possible to satisfy customer demand (see figure 2). In general, duties are straight workdays rather than split shifts.

The rostering problem is thus simple, consisting mostly of the assignment of days off. In this context there is generally no deadheading or change of terminals. In the literature, the problem is described in the context of staffing airport terminals for servicing aircraft and passengers, and in the context of freight terminals where trucks are loaded and unloaded and freight is classified. It is similar to the well-known telephone operator scheduling problem [50]. Two types of workers are often considered in this context, regular (8 hours per day) and part-time (4 hours per day). Often, no overtime is considered and part-time workers are not allowed to work more than four consecutive hours per day. Starting times for work schedules may be restricted (for example, no schedule can start between 00h00 and 06h00) or limited in number (no more than five different shifts).

SCHEDULING STRAIGHT SHIFTS WHEN DEMAND FOR SERVICES VARIES WITH TIME OF DAY

PROBLEM CONTEXT-As mentioned earlier, this problem arises mostly in the context of terminal operations. It has been described mostly for airport counter staffing and airport cleaning and service staff scheduling [2, 16, 20, 42]. From an airline schedule, the levels of personnel necessary to service customers and aircraft can be deduced. From this service curve level, a work schedule for staff must be developed. The same problem arises in freight terminals where unloading and sorting of arriving truck loads must be done in time for the loading of scheduled departure trucks [45]. In urban transit the problem may arise in subway stations or bus terminals for staffing ticket counters; however split shifts may also be permitted in this case. Another example is in the context of bus operations when drivers must all perform straight shifts (e.g. Laval, Québec).

Other applications include highway patrol, telephone operators, traffic office staff, supermarket staff, turnpike toll collecters, refuse collecters, etc.

PROBLEM FORMULATION [23]-Two common characteristics of many of these applications are:

1) a demand for immediate service that is difficult to defer or backlog;
2) variations in demand throughout the course of a day or week;
3) most of the time a fixed location.

In order to efficiently schedule workers to meet irregular demand patterns it is usually necessary to use a variety of work shift patterns which are allowed to overlap in order to create a worker surplus profile similar to the demand profile.

In this section we only consider the use of straight shifts, i.e. shifts with small breaks for coffee or lunch, generally paid.

Mathematically, the problem can be formulated as follows. This formulation has been suggested by Dantzig [18] and Segal [50] and others. It involves the minimization of the total cost of an operator shift schedule subject to meeting the demand in each period:

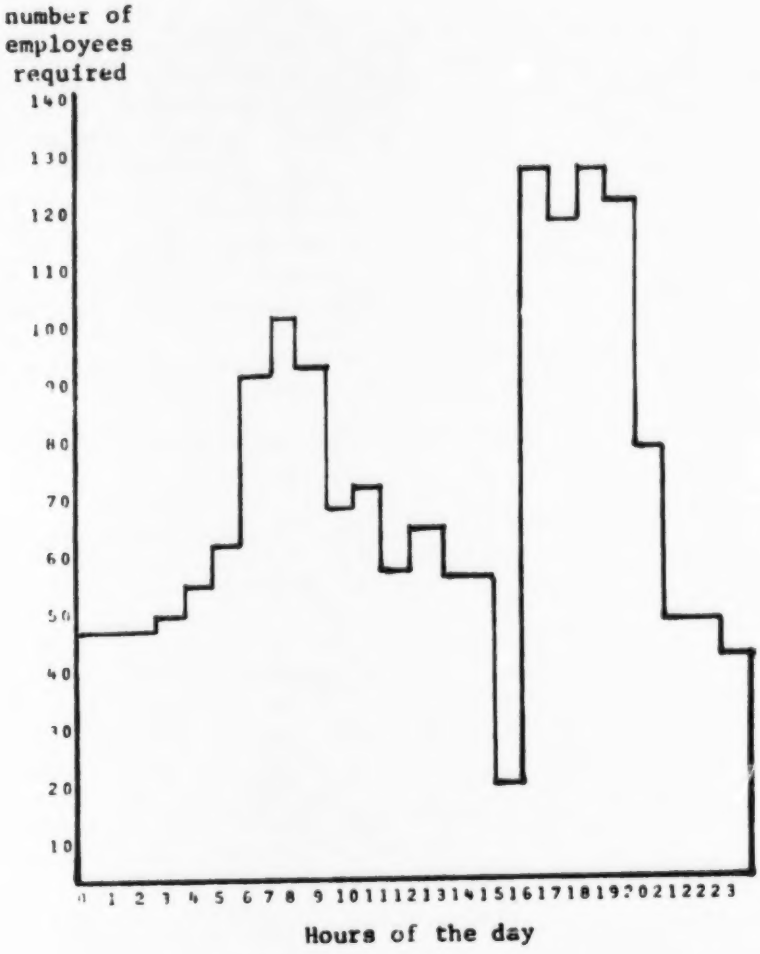


Figure 2 - Distribution of the number of employees required in a typical freight terminal [45]

$$\text{Min } Z = \sum_{j=1}^{n} c_j x_j \qquad (1)$$

Subject to

$$\sum_{j=1}^{n} a_{tj}\, x_j \geq d_t, \quad t=1,2\ldots,m$$

$$x_j \geq 0, \text{ integer}, \quad j=1,2,\ldots,n$$

Where :
- m = the number of periods to be scheduled
- n = the number of different workshift types
- x_j = the number of operators to be assigned to shift type j
- c_j = the cost of assigning one operator to shift type j
- d_t = the number of operators required for the t^{th} period, and
- a_{tj} = 1 if period t is a work period in shift type j; 0 if period t is an idle (or break) period in shift type j.

A modification of this formulation that allows the supply of service in some periods to drop below the requirement could also be introduced by adding slack values with penalties or intervals around the requirement value.

In the latter case the requirement constraint would become

$$d_t - s_t \leq \sum_j a_{tj} x_j \leq d_t + u_t$$

s_t and u_t are surplus or shortage tolerances.

With the inclusion of lunch breaks and coffee breaks in the formulation, the number of variables becomes too large for exact solution by integer programming on a routine basis. The number of straight shifts without any break starting on the hour or on the half hour is at maximum 48. If a one hour lunch break must be included in a nine hour workday between hour $2\frac{1}{2}$ and $5\frac{1}{2}$ after the beginning of the workday the number increases to 336 (48 x 7). Inclusion of two 15 minute coffee breaks with, say five possible positions before and after lunch will result in more than 8000 variables.

L.P. BASED APPROACH-Therefore, several non-exact solution techniques have been developed. Segal [50] has reported a two phase algorithm which first simplifies the formulation so that a network solution procedure can be used to generate an approximate solution. The second phase of the algorithm heuristically modifies the first phase solution. This approach is briefly outlined in the following section. Another approach simply uses an L.P. algorithm with a rounding algorithm to get a solution. Most authors report that the L.P. solution is often integer and that the rounding algorithm used, very often gives the optimal solution or, if not, a very good solution. Depending on how often the problem must be solved, a large I.P. code (MPSX, APEXIII) could also be used. Specialized set covering algorithms could also be adapted to this problem.

Ryan and Foster [48] discuss extensively the characteristics of the problem and its proximity to structures (referred to as constraint ordering, and unique subsequence precedence) which will make its optimal L.P. solution always integer. They also propose stronger branch and bound strategies. They note in fact that branching on a variable $x_j = 0$ (in a binary problem where each variable duty can be used only once) will rarely alter the objective function ; another variable with similar characteristics to x_j will be increased.

Given a pair of constraints (say r_1 and r_2) for which

$$0 < \sum_{j \epsilon J(r_1 r_2)} x_j < 1$$

where $J(r_1 r_2)$ is the set of all variables/duties covering both constraints r_1 and r_2 simultaneously, an effective constraint branch can be imposed by requiring that

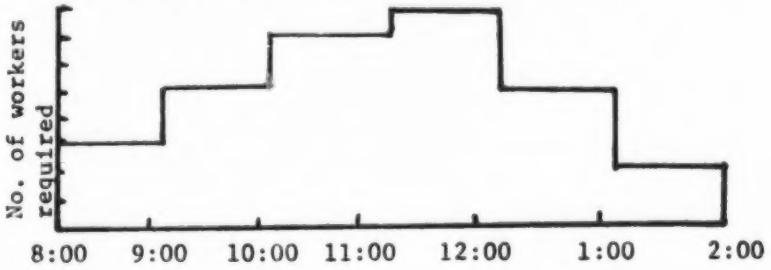
$$\sum_{j \epsilon J(r_1 r_2)} x_j \leq 0 \qquad \text{-0-branch}$$

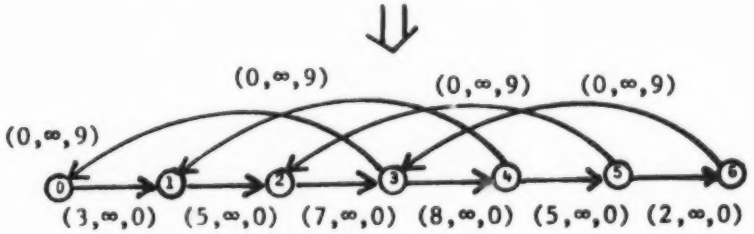
or

$$\sum_{j \epsilon J(r_1 r_2)} x_j \geq 1 \qquad \text{1-branch}$$

The structure of the tree resulting from these branching strategies is reported to be much more balanced.

A NETWORK BASED APPROACH [12]-In the case where all workdays are continuous, i.e. they do not contain lunch or rest breaks, this problem has a network formulation. The network structure arises from the fact that the columns in the matrix $A = (a_{tj})$ (problem 1) contain consecutive ones corresponding to workers working consecutive periods. We define this network which is illustrated in Figure 3 as follows. Nodes are numbered 0 to T. For each interval t an arc is drawn from node t-1 to node t. The flow on this arc is given a lower bound of d_t, an upper bound of ∞ and a cost of zero. A workday spanning time periods i through k is represented by an arc from node k to node i-1. It has a lower bound of 0, an upper bound of ∞ and a cost equal to the corresponding c_j value. A minimum cost feasible flow in this network solves the scheduling problem where the x_j values are the values of the flows in the corresponding "back arcs" from nodes k to i-1





For each arc, (lower bound, upper bound, cost) is given

The network shown assumes that the only workday possible lasts 3 consecutive hours and has a cost of 9.

Figure 3 - Formulation of Histogram Decomposition Problem as a Network Flow Problem (from Bodin et al. [12])

The major problem with this formulation is the inclusion of lunch time and coffee breaks. In his original article, Segal proposes a sequential process by which lunch and coffee breaks are assigned to periods where there is a surplus of personel. If insufficient breaks can be assigned, the curve of service demand is increased and the system reoptimized until a satisfactory solution can be found.

A computer package is available from Bell Systems to solve the telephone operator problem, for telephone companies.

HEURISTIC APPROACHES-The problem has also been solved with heuristic methods by airline companies [16,42]. For example Coubey and

Yansouni [16] describe a method by which the staff necessary to meet the workload requirements (ignoring breaks) start their shift as late as possible while still covering the workload at all times. The following phase consists of providing additional staff to allow for lunch coverage. However, prior to adding new resources to this effect, a number of tests are performed to examine all possibilities of covering lunch periods with existing staff. This is done by moving around shift starts to try to cover lunch breaks with staff surplus periods.

Finally, several authors [39,42,57] address the problem of constraining the number of possible shifts (e.g. maximum of five shifts). The L.P. formulation (problem 1) then becomes a cardinality constrained problem which is very difficult to solve optimally. The solution approach starting from the solution of the L.P. problem (problem 1) is heuristic in nature. First, shifts are moved around in order to reduce the number of shifts without increasing the manpower requirement. Secondly, if this process does not achieve the desired results, each remaining shift is dropped in turn, and the cost of using only the other shifts is evaluated. The shift which generates the smallest increase in cost is then dropped and the process is continued until the number of shifts desired is attained.

STRAIGHT AND SPLIT SHIFT SCHEDULING

In the previous context, only straight shifts were allowed. In some situations however both straight and split shifts (workdays with large unpaid breaks) are permitted. Formulation (1) is also valid in this context. We consider here in some detail a generalization of this problem that is used in urban transit for the economic evaluation of drivers' collective agreements and as a first approximation for the crew scheduling problem.

PROBLEM CONTEXT [11]-Union contract negotiations for bus driver's in transit companies have become increasingly difficult in recent years because of the complexity of the scheduling rules under negotiation and the difficulty of accurately evaluating the impact of these rules on costs. These impacts are often estimated manually, with or without the assistance of a computer, by producing a feasible schedule which incorporates proposed new scheduling rules. This process, in addition to being tedious and potentially inaccurate, cannot take into account the forecasted future development of bus services and its impact on future costs, nor can it be repeated as often as desired to study different options. Thus, decisions to accept or refuse certain scheduling rules are based partly on past experience, partly on intuition and partly on detailed scheduling. This leads companies, under political pressure during drivers strikes, to accept, without full knowledge of their cost, scheduling rules that may prove to be very expensive. The problem is then to develop an evaluation tool that can rapidly evaluate union contract scheduling rules at a tactical decision level without necessarily producing a detailed feasible solution. Such evaluations have to be made frequently in the course of negotiations. It must be a model which is easy to use, does not require too much computer time, gives good cost estimates of contract modifications, and can be adapted easily to all kinds of new rules.

THE MODEL-Strictly speaking, the problem consists of obtaining a drivers' schedule which satisfies the constraints. However, the authors chose not to try to generate a feasible schedule because of the high level of complexity and detail required. Instead, they decided to solve a relaxation of the drivers' scheduling problem on a *modified* bus schedule. The idea is to retain the essentials of the bus schedule structure and its impact on the contract rules and costs without retaining the precise bus schedule.

Firstly, *periods* defined by the user are introduced. These periods are normally 30 minutes long; they can vary, depending on the problem, between 15 and 60 minutes. The schedules of all buses are approximated in order that all buses start their block* at the beginning of a period and complete it at the end of a period. The authors further assume that relief points can only occur at the beginning of a period. Moreover, the problem is further relaxed by requiring that the working days selected be sufficient to cover the total requirement of drivers per period instead of requiring that they exactly cover all the blocks individually.

All feasible working days satisfying the union contract with one, two or three pieces of work of an integer number of periods are generated. The generating rules eliminate, under instructions from the user, working days which are of no interest or which do not correspond to unwritten rules or common practice (for example, a split-shift with a break during peak hour). All the working days are costed according to the union contract under study. The linear program then searches for the set of working days that will provide the number of drivers required during each period (as problem 1) and satisfy other additional constraints at minimum cost.

Two types of additional constraints are added. The first type is related to the union contract. All the constraints related to the total number of working days of a certain type are considered. These include constraints on the minimum or maximum number of straight shifts and constraints on the maximum number of working days with certain undesirable characteristics like long spread, small breaks, etc. The second type of constraints (feasibility constraints) was added after the first tests were done. It was realized that the solution obtained was difficult to convert into a practical solution because the linear program was assuming that a small vehicle block (i.e. vehicle block less than four hours) could be split: in practice, where a piece of work must last at least two hours, these small blocks are nearly always assigned as a piece of work to one driver. It was thus decided to add a series of constraints to represent this fact (constraint (iii) in next section). These constraints require at least as many working days with a piece of work from period k to period h as there are small blocks from period k to period h and this for all combinations of k and h, for which there exists a small block. For the same purpose, another set of constraints (constraints (iv) in next section) was added to generate as many pieces of work starting in a period as there are long blocks beginning in the given period. A detailed formulation of the problem follows.

---

* A block is a set of runs that a bus covers between its departure and return to the garage.

A MATHEMATICAL FORMULATION OF 'HASTUS-MACRO'*-For simplicity of presentation, we now assume that there are at most two pieces of work in a working day. The extensions to three or more pieces are immediate. The basic variables are :

x_{ijkh} : a variable corresponding to the number of drivers assigned to a workday composed of a piece of work from period i to j and a piece of work from period k to period h. If k=h=0 the workday does not have a break. It could be either a straight run or a tripper (leftover piece of work done in overtime).

The model is :

$$\text{Min} \sum_{(i,j,k,h,)\in X} c_{ijkh} x_{ijkh} \qquad (2)$$

$X = \{(i,j,k,h) \mid x_{ijkh}$ corresponds to a feasible workday generated$\}$

c_{ijkh} : cost of the workday corresponding to x_{ijkh}.

s.t.

i) A set of constraints ensuring that during all periods of the day the number of drivers working is greater than or equal to the number of vehicles in circulation.

$$\sum_{\substack{(i,j,k,h,)\\ \in X(t)}} x_{ijkh} \geq n_t \qquad \forall\ t$$

where $X(t) = \{(i,j,k,h) \mid i \leq t \leq k \quad \text{or} \quad k \leq t \leq h\}$
and n_t is the number of buses in operation at time period t.

ii) Union contract constraints as previously described

$$Ux \leq u$$

where x is the vector with components x_{ijkh}.

iii) The number of pieces of work from p to q is at least as large as the number of small blocks from p to q.

$$\sum_{\substack{(i,j,k,h)\\ \in F_{pq}}} x_{ijkh} \geq Q_{pq} \qquad \forall\ (p,q) \text{ for which a small block exists}$$

where $F_{pq} = \{(i,j,k,h) \mid (i,j) = (p,q) \text{ or } (k,h) = (p,q)\}$
and Q_{pq} is the number of blocks from p to q that must correspond to a piece (p,q).

iv) The number of pieces of work beginning in t must at least be equal to the number of blocks starting in t

$$\sum_{\substack{(i,j,k,h)\\ \in L(t)}} x_{ijkh} \geq K_t \qquad \forall\ t$$

where $L(t) = \{(i,j,k,h) \mid i = t \quad \text{or} \quad k = t\}$
K_t : the number of blocks beginning in period t.

v) $x_{ijkh} \geq 0$. The integrality of x_{ijkh} has been relaxed.

---

* Acronym for Horaire et Assignation pour un Système de Transport Urbain et Semi-urbain - approche macro.

SOLUTION ALGORITHMS-Even if the x_i should be integer, the authors chose to solve the problem using a continuous L.P. procedure. Because of the number of variables and constraints an exact I.P. algorithm would use too much computer time and be too costly for a program designed to be used repetitively. Moreover the computer run for a set of working rules and costs to be tested are always compared to a basic run (one using actual working conditions). Under these conditions, the authors preferred to compare the relative cost of the test runs against the basic run using a straight L.P. rather than using an approximate I.P. procedure.

Obtaining the solution of the L.P. problem is straightforward. At first, standard L.P. packages such as APEXII and MPSX were used but were found cumbersome and costly for users. Because the problem rarely has more than four thousand variables and three hundred rows, an L.P.subroutine from the Land and Powell [29] package was adapted. As in any linear program, however, the problem lies not in the solution technique, but in data preparation and solution interpretation. Even if the system was planned to be used by persons with some technical skill in quantitative methods, it was essential to develop a fairly natural language to allow the users to easily specify the problem and constraints. Such a package was developed and named HASTUS-macro. A full description of this approach together with its use is given in Blais and Rousseau [11].

GENERALIZATION-From the structure of the problem just described, it is easy to use the approach and even the corresponding package for the scheduling of both straight and split shifts in terminal operation. The system could also be used, and was in fact used in Québec City for a while, to generate approximate schedules for bus crews that were later manually converted into operational schedules. This approximate solution is also used as initial input to another system which automatically computes a feasible solution to the scheduling problem as we will see later on.

SET COVERING FOR IN-VEHICLE CREW SCHEDULING

EXACT SET COVERING-Crew scheduling in any context can always be formulated as a set covering (or set partitioning) problem. The formulation is well-known:

$$\text{Min} \sum_{j=1}^{n} c_j x_j \qquad (3)$$

$$\text{s.t.} \sum_{j=1}^{n} a_{ij} x_j \geq 1 \qquad i=1,\ldots,m$$
$$(=)$$

$$x_j = 0,1$$

where c_j is the cost of workday j and $a_{ij} = 1$ if workday j covers task i. All tasks i (i=1,...,m) must be performed. A task corresponds to a vehicle trip between two relief points (flight segment, bus trip, train trip) where the crew could be changed. The set covering formulation is used when two crews can be on the same vehicle, one deadheading to another location to take on another assignment. The set

partitioning formulation is used when deadheading is not permitted.

When the number of variables is fairly limited (several thousand variables rather than several hundred thousand) several solution approaches are possible.

In her solution approach, Heurgon [24,25] uses a simple branch and bound algorithm and notes that most solutions were integer after the L.P. solution. Marsten [34], for airline crew scheduling uses SETPAR, an algorithm he developed for large set partitioning problems. (Table 2 from Masrten et al. [35] reports good results for his algorithm on the crew scheduling problem.) Balas and Ho [7] have also developed a special algorithm for large set covering. Salkin's heuristic [30,49] which completes the solution of the L.P. to an integer solution reports that his algorithm gives solutions diverging less than 1% than the optimal solution. Finally Etcheberry [19,21] uses Lagrangian relaxation and subgradient optimization in Masrten branch and bound strategy. Shepardson [35,52] is using a special set partitioning formulation for a specific single line transit crew scheduling problem.

TABLE 2 - Set covering for airline crew scheduling using SETPAR on IBM 370/148 [35]

Rows	Columns (pairings)	Nb of int. sols	Savings (%) over manual sols	Generation time (min)	Solution time (min)
126	28,934	7	5.34	5[illegible].73	1304.93*
129	23,226	2	12.36	2[illegible].45	75.63
92	24,428	4	16.39	29.40	317.72
113	14,651	2	27.90	23.63	240.45
103	5763	1	27.98	27.58	18.25

* optimization not completed

He notes that the set partitioning problem (3) for a single line transit scheduling problem sometimes has the following characteristic: each column has two strings of consecutive ones, a string for each piece of work. This occurs in the case where union regulations require that a driver works on only one bus in each of his two pieces of work and the number of pieces is limited to two.

It is easy to show that a set partitioning problem in which each column has exactly *one* string of consecutive ones is equivalent to a shortest path problem.

This suggests the reformulation of the problem as a shortest path problem with side constraints [35]. Each column that has two strings of ones A_j is replaced by a column A_j^1 that has only the first string and a column A_j^2 that has only the second string. The variable X_j is replaced by two variables, X_j^1 and X_j^2, and c_j is replaced by any two numbers c_j^1 and c_j^2, where

$$c_j^1 + c_j^2 = c_j, \quad \text{e.g.} \quad c_j^1 = c_j^2 = c_j/2.$$

$$\text{Min} \quad \sum_{j=1}^{n} (c_j^1 X_j^1 + c_j^2 X_j^2), \tag{4}$$

$$\text{subject to} \quad \sum_{j=1}^{n} (A_j^1 X_j^1 + A_j^2 X_j^2) = e,$$

$$X_j^1 - X_j^2 = 0, \qquad j = 1,\ldots,n,$$

$$X_j^1, X_j^2 = 0 \text{ or } 1, \qquad j = 1,\ldots,n,$$

where e is an m vector of ones. This is a shortest path problem with side constraints $X_j^1 = X_j^2$ requiring that the two parts from the original X_j variable be selected or rejected together. This immediately suggest to the authors Lagrangian relaxation, moving the side constraints into the objective function with appropriate Lagrange multipliers.

A specialized branch-and-bound algorithm based on the Lagrangian relaxation outlined above and using subgradient optimization on the dual problem is presented in Shepardson [52].

The method of solution gives results in much less time than the utilization of the set partitioning package SETPAR (See Table 3).

TABLE 3 - (from Marsten and Shepardson, [35]). Comparison between the Set partitioning algorithm (SETPAR) and the subgradient approach (Helsinki) (SO is the best value obtained by subgradient optimization at the top of the tree)

No.	Rows	Columns	SO	LP	IP	Gap(%)	HELSINKI (sec)	SETPAR (sec)
5-1	57	551	669	696.6	802	15.1	26.3	71.2
5-2	78	525	1555	1576.8	1637	3.8	87.7	>84[a]
5-3	73	412	863	887.6	1294	45.8	174.3	b
5-4	60	411	449	457.7	511	11.6	15.7	50.0
5-5	78	1,103	7382	-	8755	≤18.6	86.5	b
5-6	156	11,152	4501	-	7340	≤63	3720	b

[a] Problem abandoned after 84 sec. best solution found was 1893.

[b] Problem not attempted.

However, the main problem in using a set covering approach for crew scheduling problems is generally the large number of possible workday variables. In air crew scheduling, several million variables are possible and there may be even more in a transit problem where interlining is permitted. Several heuristics have been constructed using a set covering or partitioning based approach.

RUBIN'S ALGORITHM-Rubin's algorithm [46,47] for air crew scheduling is probably the most used system by large airlines, because it has been integrated into a comprehensive user oriented package (IBM and American Airlines).

It is a marginal improvement heuristic. It requires a starting solution normally provided by the scheduler. The algorithm is aimed at improving this initial solution. For every subset S_ℓ of pairings considered, it creates a small set covering problem on the subset of tasks covered by S_ℓ. For this small set covering problem it is possible to generate all possible variables covering the subset of tasks (including pairing in S_ℓ). These problems are small and thus easy to solve optimally. If the solution generated has a better cost than S_ℓ, the solution replaces S_ℓ to become the new starting solution. First, only subsets of one pairing (level 1) are considered by Rubin until no improvement has been generated for a full cycle through the variables. Then subsets of two pairings (level 2) are considered followed by subsets of three pairings (level 3), etc. The algorithm is stopped on a time limit. Rarely will the algorithm have the time to consider subsets of four pairings and more. The choice of a suitable strategy for considering subsets of pairings that can conceivably be improved is critical for the algorithm (e.g. after an improvement at level k, Rubin goes back to level k-1). Some details are given in Rubin's original paper. It is believed that the American Airlines package based on this approach uses another set of strategies. It is generally recognized that only 1% of the trials for improvement are successful and that the quality of the solution produced depends heavily on the quality of the initial solution given.

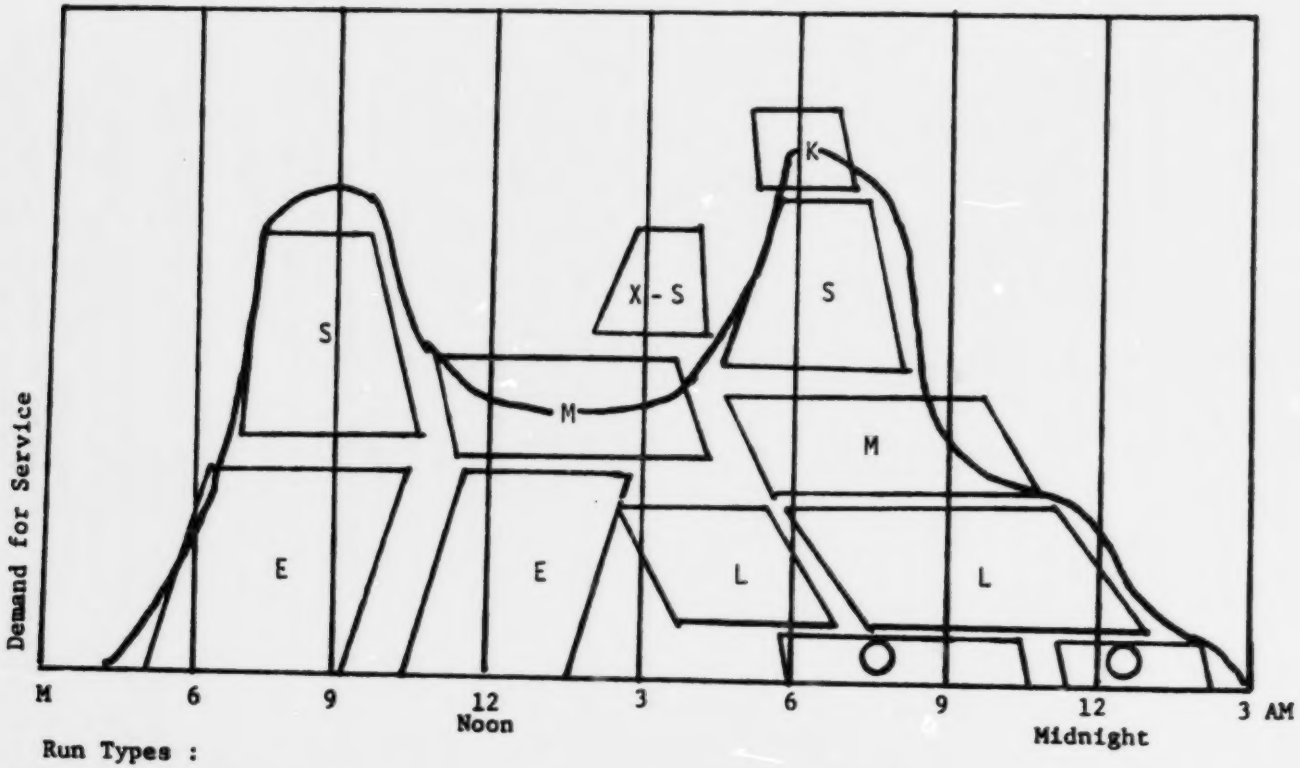
RESTRICTIVE APPROACH [12]-Another approach to the problem of too many variables is to use a set of heuristics to limit the generation of variables or to decompose the problem into subproblems which are to solve optimally. In urban transit crew scheduling, several examples of such algorithms exist.

To understand these approaches we must reexamine the service profile graph given in Figure 4. The decomposition approach of Ward et al. [55] classifies runs according to where their component places fit into the service profile. Then they solve a sequence of set partitioning problems with the first problem generating the workdays that end the latest (owls), the second problem generating the workdays that end the second latest (late straights) and so on until all trips are covered. Of course, the set of colums generated at any step will be constrained by the workdays generated during previous steps. Since all columns are not considered simultaneously, the approach cannot guarantee an optimal solution to the set partitioning problem.

Parker and Smith [43] include all trips in the service profile within one set covering problem, however, they use a variety of criteria to restrict the set of colums considered. These include:

a) lower and upper bounds on piece lenght,
b) lower and upper bounds on length of breaks,
c) restriction on piece start and end times based on run classification similar to the one shown in Figure 4.

It is widely felt that the service profile contains valuable information that should be used in generating crew schedules. These two approaches attempt to use this information. The approach of Lessard et al. [31], which is described later in detail, includes profile information within the crew scheduling algorithm by solving the approximation problem described earlier (HASTUS-macro). The two set partitioning/covering methods just described have undergone limited computational testing and the results appear promising. However, they



Run Types :

E - Early Straight Runs
M - Middle Straight Runs
S - Split Runs
X - Extra pieces (for use as Trippers, i.e., short one-piece runs, or the 3rd piece of work on a straight run)
L - Late Straight Runs
O - Owls (late late straight runs)

Figure 4 - Service Profile and Decomposition into Run Types [12]

have not yet been operationalized.

COLUMN GENERATION-Finally, for airline crew scheduling, it is also possible to use a column generation algorithm to generate the pairings when necessary. This algorithm, proposed by Crainic [17], first generates heuristically a set of pairings. Then the set covering problem is solved linearly and the dual variables are used to generate a set of other pairings with a negative reduced cost that are added to the original problem. This process is pursued until improvement in the objective function becomes small. Then Salkin's heuristic [30,49] is used to obtain an integer solution. Pairings with negative reduced cost are generated using a large set of duty periods generated with limiting constraints. Partial reduced costs are computed for these duty periods with the dual variables obtained from the set covering problem. The cost function of the crew schedule is such that the reduced cost of a pairing is greater than the sum of the partial reduced cost of the duty period.

HEURISTIC SET PARTITIONING FOR AIRLINE CREW SCHEDULING [12]

Heuristics are the only recourse for most of the larger crew scheduling problems encountered in practice. Baker [4] and Baker, Bodin and Fisher [5] describe the use of heuristic set covering algorithms for solving air crew scheduling problems. These algorithms together with solution improvement methods which are described later on, are being used by the Federal Express Corporation for both single and multiple depot crew scheduling. Similar approaches have been used by the Selective Bidding Corporation. These algorithms are quite fast and have been compared favorably with competitive procedures.

The Federal Express crew scheduling system contains three steps:

i) a pairing generator used to enumerate candidate crew schedules,
ii) a heuristic set covering algorithm, used to obtain an initial covering solution,
iii) several solution improvement techniques used to reduce the total cost of the covering solution.

The pairings generator enumerates new pairings which are feasible and computes the cost of each pairing. In this enumeration, subjective reduction procedures are incorporated to limit the number of pairings generated and to reduce the computer run time of the generator module. The imposition of subjective reduction procedures in the enumeration process was used to enforce desired operational policies associated with the crew pairings (e.g., maximum time away from base, maximum layover time, maximum time away from base to flight time ratio, etc.).

Four distinct heuristics for finding feasible solutions to the set covering problem were evaluated. The first procedure termed Algorithm A, begins with an empty solution set and adds to the solution the crew schedule that minimizes the cost per uncovered flight. The procedure continues until all flights are covered. The second procedure, Algorithm B, considers the crew schedules in ascending order of cost per flight leg covered. The algorithm begins with an empty solution set and adds the first crew schedule from the sorted list that contains no previously covered flights. If a solution is

not obtained after a complete pass through the crew schedules, the process is repeated, allowing candidate crew schedules to contain one previously covered flight. This process is repeated, allowing increased overcoverages until a complete solution is obtained.

The third procedure considered, Algorithm C, was proposed by Johnson [28] for the case where all $c_j = 1$; this gives the minimum number of pairings needed to cover all the legs. Algorithm C works in a manner identical to Algorithm A except that crew schedules are selected based on the maximum number of uncovered flights. The rationale behind this approach is that if the column generator contains subjective tests that produce only efficient crew schedules, then a minimum set of such schedules may produce a near-optimal solution. The fourth procedure, termed Algorithm D, is Johnson's technique for the exact covering problem. This procedure starts with an empty solution set and adds to the solution the crew schedule that minimizes the ratio of covered to uncovered flight segments. The rationale behind this algorithm is similar to that of Algorithm C.

Baker experimented with several data bases from the Federal Express Corporation. These results are presented in Table 4.

The computational results indicate that Algorithm A, the selection of crew schedules according to a minimum cost per uncovered flight leg criterion, dominates the other set covering heuristics considered. In the test case with the fewest columns, Algorithm A identified the optimal solution. The worst performance reported for Algorithm A was a 10.2% deviation in test case 4. Additional computational evidence in support of Algorithm A as an effective set covering heuristic may be found in the recent papers of Balas and Ho [7]. In addition, in Baker [4], discussions of worst case bounds for set covering heuristics may be found.

Baker, Bodin, Finnegan and Ponder [6] and Baker, Bodin and Fisher [5] propose three solution improvement procedures : an extended 2-opt algorithm, a solution merge technique and a hub turn determination procedure. A brief description of these solution improvement procedures is given below.

The extended 2-opt procedure which assumes a single feasible solution to the set covering problem as input is an adaptation of the 2-opt procedure proposed by Lin [32] for the traveling salesman problem. In this interpretation of the air crew scheduling problem, each flight leg to be scheduled is viewed as a city to be visited. The exchange of arcs connecting the cities on a traveling salesman's route is analogous to changing the sequence in which the flight legs are scheduled. Since any such exchange must take place at a stop common to all the candidate pairings, each city is examined for possible exchanges of pairings completions at that point. If k pairings transit a particular city, then a kxk assignment problem may be solved to determine the optimal reconfiguration of the crew schedules.

The second solution improvement procedure employed was a solution merging technique. Input to the merge procedure is a set of feasible solutions. Solution merging, which was originally proposed by Hinson and Mulherkar [26] for the Federal Express aircraft routing problem, exploits the availability of feasible covering solutions provided by the set covering heuristics. The first solution found becomes the incumbent. As subsequent solutions are found they are

compared with the incumbent to determine if a reduced cost composite solution can be formed. Since this technique compares only two solutions at a time, the determination of the lower cost composite solution can be made very efficiently. The merge problem can be viewed as a restricted set partitioning problem where the rows are the legs and the columns are the pairings from the two solutions. This problem is restricted in the sense that only a fraction of the set of all possible columns were considered.

The final solution improvement technique employed was the joining of two pairings together to find a pairing which is hubturned. A hubturn occurs when a crew transits the crew base on a pairing. All pairings start and end at the crew base. It is often possible to exploit the nonlinearity of the crew scheduling objective function by allowing two individual pairings to be performed sequentially as a reduced cost hubturn pairing. A matching algorithm can be used to find a hubturned solution. The nodes in the matching problem are the pairings in the best solution found so far and the arcs represent joining two pairings into a pairing with a hubturn. The costs on the arcs are the costs of corresponding pairings without hubturns.

TABLE 4 - Computational Results for Set Covering Heuristics [12]

				Algorithm			
No. Hubs	Problem	Rows	Columns	A*	B**	C*	D*
1	1	376	1245	0 (2.15)	.01 (1.01)	14.4 (2.20)	14.5 (2.20)
1	2	426	2780	1.6 (5.60)	4.1 (2.96)	49.5 (5.71)	13.7 (5.46)
2	3	180	3431	3.8 (6.30)	8.1 (4.10)	36.7 (6.20)	20.1 (6.20)
4	4	161	4565	10.2 (9.35)	22.0 (7.50)	50.3 (9.29)	23.6 (8.86)
5	5	335	5290	6.7 (8.76)	8.3 (8.61)	62.5 (7.70)	18.3 (8.53)

Entries in columns A, B, C and D give percent deviation from the optimal solution and (in parentheses) minutes of processor time on a Burroughs 6700 mainframe.

* Best of 10 solutions / ** Best of 100 solutions

Computational results are presented in Table 5 for several multiple hub/multiple domicile network configurations proposed by the Federal Express Corporation using these improvement procedures. The data sets considered represent various domicile combinations of interest to Federal Express Flight Operations for the two-, four-, and five-hub systems. In each case the solution procedure involved the generation of candidate crew schedules subject to various subjective and objective reduction procedures, the selection of a set

of initial solutions using a set covering heuristic (Algorithm A), and the improvement of the initial solutions by the extended 2-OPT, MERGE, and hubturn procedures. For each phase of the solution process the objective function value, in total credit minutes, and the processor time, in minutes and seconds on the Burroughs 6700 computer, are reported. Data set 1 was benchmarked on the CAPS system, the Continental Airlines pairings generator system. The final solution from the CAPS system did not satisfy all the constraints on the pairings Federal Express required, and hence, represented a lower bound rather than a feasible solution to the problem.

This algorithm has not been tested on any data base other than those created by Federal Express Corporation. Because of the variety of cases run (see Table 5) the authors believe it can be utilized for other airlines with appropriate modifications to take into account constraints specific to the airline.

DECOMPOSITION APPROACH FOR TRANSIT CREW SCHEDULING

The most popular approach for transit crew scheduling proceeds as follows.

First the blocks are split into pieces of work. Secondly, the pieces of work are matched together to form workdays and thirdly the solution is improved heuristically and manually. A matching or an assignment problem formulation for the formation of the workdays is generally used. This will be discussed more formally in the presentation of the HASTUS model in the next section. Various strategies are possible for splitting the blocks into pieces of work. Most approaches use heuristics adapted to the type of problem considered. In RUCUS [56], the first general package developed for transit crew scheduling, a set of parameters enable the user to partially determine the method by which the pieces will be cut. In Sage runcutting and RUCUS II, an interactive system for transit crew scheduling, the user has direct control on the ways the pieces are cut. The Hamburg system [27] and Task's system (from A. Wren), include heuristic approaches adapted to their respective problems related to German and British working conditions. The HASTUS system uses a relaxation of the problem to determine how to cut the pieces. Ball et al. [8] form the pieces and blocks by sequentially matching the trips. These last two methods are now described in some detail. First we recall some terminology.

A block is the itinerary of a vehicle between its departure from and its return to the garage. It includes all deadhead times to take the bus out of and into the garage to and from the route(s) it services. There are generally short blocks to cover the peak periods and long blocks for the basic service.

Relief times are the times corresponding to places on a route where a change of drivers is possible. In general the number of such places is small.

A piece of work refers to the period of time during which a driver works continuously with the same vehicle without a break. Generally the number of pieces of work in a workday is limited (2, 3 or 4) and the pieces must have a minimum duration (2-3 hours).

A workday (run) is a daily assignment consisting of one or more pieces of work, which must satisfy the union contract rules.

TABLE 5 - Computational Results for Heuristic Set Covering and Solution Improvement Methods [12]

Data set	Hubs (Dom)	Flight legs	Pairing Gener.	Best sol. Algorithm A among 10 sol.	Ext 2-opt	Merge for 20 sol.	Hubturn	Other	Benchmark
1	2 (1)	426	2780	37 291	37 037	36 892	27 939	27 335	27 878
2	4 (2)	180	506	27 463	27 404	27 404	27 250	27 250	27 889
3	5 (5)	335	5290	51 217	50 859	50 767	49 250	48 862	53 531

A tripper is a small piece of work which is normally done in overtime by regular drivers or assigned to stand-by drivers. In general the companies hope for a few or no trippers. By extension and simplicity of notation we consider that a tripper is a workday. However we assume that no explicit bound or constraint on the number of trippers is given which always assures us of the existence of a feasible schedule.

THE "HASTUS" SYSTEM [15,31]

THE MODEL.-For simplicity of presentation, we now assume that there are at most two pieces in a workday. The extension to three or more pieces of work is simple. In this section a general formulation of the problem is introduced. This generality is restricted only by the fact that we assume that the minimum break required by the union in a workday is always sufficient to allow for driver travel time between two pieces of work. If this is not the case, our solution strategy is not modified ; but this fact must be taken into account when the pieces of work are matched together to form workdays.

The notation required for the HASTUS model is first introduced. We denote by a pair (i,j) a piece of work starting at time i and ending at time j. Only feasible pairs (i,j) are considered, that is pairs such that both i and j correspond to either a starting time, ending time or relief time in a given block. Note that (i,j) may be feasible relative to several blocks. For practical reasons, we also consider in the feasible set of pieces the null piece (0,0).

A quadruplet (i,j,k,h) denotes a workday made up of the feasible pairs (i,j) and (k,h). Only workdays (i,j,k,h,) which are compatible with the union contract and the company regulations are considered. If either (i,j) or (k,h) is a null piece, the workday is either without break or corresponds to a tripper.

In addition we define :

L	: the number of blocks.
K	: the number of distinct pieces.
I	: the set of feasible quadruplets (i,j,k,h).
$I(i,j) \subset I$	: the set of quadruplets where one of the pieces is (i,j).
T_ℓ	: the set of all times k which are either relief times, starting times or ending times for block ℓ.
x_{ijkh}	: an integer non-negative variable corresponding to the number of drivers assigned to a workday (i,j,k,h).
x	: the vector with components $x_{ijkh}, (i,j,k,h) \in I$.
c_{ijkh}	: the cost of the workday (i,j,k,h) ; this is normally the direct cost according to the union contract.
y^ℓ_{ij}	: a binary variable taking value 1 if a piece (i,j) is used on a block ℓ to be part of a driver workday.
y^ℓ	: the vector with component y^ℓ_{ij} for *all* feasible (i,j).
y	: the vector with component $y^\ell, \ell=1,..,L$.
Y	: the set of all y such that $y^\ell_{ij}=0$ if $i \notin T_\ell$ or $j \notin T_\ell, \ell = 1,2,..,L$.

The problem can now be formulated as follows :

$$\text{Min} \sum_{l} c_{ijkh}\, x_{ijkh}$$

$$\text{s.t.} \quad \text{i)} \sum_{l(i,j)} x_{mnkh} - \sum_{\ell} y^{\ell}_{ij} = 0 \qquad \text{for all feasible } (i,j)$$

$$\text{ii)} \quad Dx \geq d \qquad (5)$$

$$\text{iii)} \sum_{i \in T_{\ell}} y^{\ell}_{ik} - \sum_{j \in T_{\ell}} y^{\ell}_{kj} = b^{\ell}_{k} \qquad \text{for all } k \in T_{\ell}, \ \forall \ \ell ,$$

$$\text{where } b^{\ell}_{k} = \begin{cases} -1 \text{ if } k \text{ is the starting time of block } \ell \\ +1 \text{ if } k \text{ is the ending time of block } \ell \\ 0 \text{ otherwise} \end{cases}$$

$$\text{iv)} \quad x \geq 0 \text{ and integer}$$
$$y \in Y$$

The constraints (iii) correspond to the flow formulation used to partition each block ℓ into pieces of work. Figure 5 illustrates the concept. The feasible pieces of work (we are assuming here a minimum length of two hours) are represented by arcs and listed in the figure. We have assumed for simplicity that a relief point exists every hour in this example. The indices k in the constraint formulation correspond to points where a change of drivers is feasible and the constraints ensure that a flow of one goes from the origin of the block (7:00) to its end (13:00), using arcs (i,j) corresponding to feasible pieces. Constraints (iii) define a flow problem on an uncapacitated network.

By constraints (i) we ensure that any feasible piece (i,j), used to partition a block will be used in a workday of type (i,j,k,h) or (k,h,i,j,).

feasible pieces are

7 - 9	7 - 11	9 - 11	10 - 13
7 - 10	7 - 13	9 - 13	11 - 13

Figure 5 - The flow formulation for the partition of a block

Constraints of type (ii) refer to any particular constraint which might arise in real life problems. Most of the time, these constraints take into account particular clauses of the union contracts like a bound (either lower or upper) on the number of workdays without break or a bound on the average length of the workdays.

Unless the problem is small (i.e. the number of blocks is small and thus the number of pieces and workdays is not too large), this

formulation seems to be impractical. Given a medium-sized problem like that of Québec City or Florence, we can easily generate over five thousand pieces and one million workday variables without considering the flow variables and the difficulty of solving the x_{ijkh} in integer form. In fact this formulation is nearly equivalent to the set covering formulation. However, the formulation proposed (5) seems to be more amenable to a solution strategy that can handle very large problems. The problem can be stated in a more compact form as :

$$\begin{aligned} &\text{Min } cx \\ &\text{i) } B\,x - \sum_{\ell} y^{\ell} = 0 \qquad (6) \\ &\text{ii) } D\,x \geq 0 \\ &\text{iii) } E\,y^{\ell} = b^{\ell} \qquad \ell=1,\ldots,L \\ &\text{iv) } x \geq 0, \quad y \in Y \end{aligned}$$

The definition of c, B, E and B^{ℓ} follows from formulation (5) of the problem.

SOLUTION STRATEGY- The chosen solution strategy is to use the obvious decomposition of the problem into the generation of a partition of each block into pieces of work (constraint (iii)) and the matching problem (constraint (i)) to form workdays. It has to be ensured that constraints (ii) are also satisfied. Three main steps compose this strategy:

Step 1 : Using a relaxation of the whole problem we generate a partition of each block into feasible pieces of work that will respect as much as possible the constraint set (ii).

Step 2 : We solve the matching problem to obtain a solution to the bus driver scheduling problem.

Step 3 : Using a set of heuristic techniques, the solution previously obtained is improved while ensuring that constraint set (ii) is respected.

PARTITIONING THE BLOCKS-In the HASTUS system, the relaxation (HASTUS-macro) described earlier is used to help generate a first feasible solution as close as possible to the lower bound it indicates. This is done first by generating an initial block partition that uses similar types of pieces of work and in approximately the same number as indicated by HASTUS-macro (2).

The following problem is considered :

$$\begin{aligned} &\text{Min } (B\bar{x} - \sum_{\ell}(y^{\ell}))^{2} \\ &E\,y^{\ell} = b^{\ell} \qquad \ell=1,\ldots,L \qquad (7) \\ &y \in Y \end{aligned}$$

$$\text{where } (w)^2 = \sum_i w_i^2 \quad w \in R^n$$

$\bar{x}$ is the optimal solution of the HASTUS-macro problem. For simplicity of notation we consider for the moment that the pieces of work are defined on periods as in the HASTUS-macro problem. Problem (7) can be

easily solved using a heuristic procedure. In fact, if all but one of the decision vectors y^ℓ are fixed ($y^\ell = \bar{y}^\ell$ $\ell \neq r$), the nonlinear objective function of (7) becomes a linear function of the elements of the remaining decision vector, y^r. Such a function is :

$$(B\bar{x} - \sum_{\ell \neq r} \bar{y}^\ell)^2 + (e - 2(B\bar{x} - \sum_{\ell \neq r} \bar{y}^\ell))y^r$$

where e is a vector of ones and use is made of the equality $(y^r_{ij})^2 = y^r_{ij}$. Note that the constraints of the problem in the y^r variable vector are those of the shortest path problem. We define

$$V^r(\bar{x},\bar{y}) = (e - 2(B\bar{x} - \sum_{\ell \neq r} \bar{y}^\ell))$$

Then, in order to solve (7), we solve a sequence of shortest path subproblems P_r :

$$P_r : \text{Min } V^r(\bar{x},\bar{y})y^r$$

$$\text{s.t. } E\, y^r = b^r$$

$$y^r \in Y$$

The algorithm MINSQ which finds a suboptimal solution to (7) is described below :

MINSQ

1. a) Take any feasible solution $\bar{y}_o$ and evaluate

$$Z^0 = (B\bar{x} - \sum_\ell \bar{y}^\ell_o)^2.$$

   b) $k \leftarrow 1$.

2. a) Solve P_ℓ with $\bar{y} = \bar{y}_{k-1}$, for $\ell = 1,..,L$; let $\bar{y}^\ell_k, \ell = 1,..,L$ be the solution obtained.

   b) Evaluate $Z^k = (B\bar{x} - \sum_\ell \bar{y}^\ell)^2$.

3. a) If $Z^k = Z^{k-1}$, stop.

   b) $k \leftarrow k+1$ go to 2.

Let $\tilde{y}^\ell$ ($\ell = 1,..,L$) denote the solution obtained for problem (7). This solution satisfies the constraints (iii) of problem (6).

In reality when this algorithm is applied, the objective function $(B\bar{x} - \sum y^\ell)^2$ uses pieces defined on period while the shortest path problem uses pieces defined on real time. For the evaluation of the objective function we transform the pieces defined on real time into pieces defined on period. When the algorithm stops we thus have a partition of each block into pieces of work. The next step consists in building up a first feasible solution.

THE MATCHING PROBLEM-A maximum weight matching problem can be set up to generate the best set of workdays with a minimum number of trippers. This problem can be defined as follows :

$$\text{Max} \sum_{I'} \bar{c}_{ijkh} x_{ijkh} \qquad (8)$$

$$\sum_{I'(ij)} x_{mnkh} \leq 1 , \qquad (ij) \in V$$

$$x_{ijkh} = 0,1$$

where $\bar{c}_{ijkh} = M - c_{ijkh}$

V : the set of feasible pieces defined in real times resulting from the partitioning of the blocks

I' : the set of feasible workdays using pieces from V

I'(ij) : the set of feasible workdays (mnkh) $\in$ I' where one of the pieces is (ij)

M : a large number ; it corresponds to the relative penalty associated with a tripper.

Note that contrary to problem (5) only the x_{ijkh} which are feasible and use pieces of work previously generated by the partition algorithm are used here.

A flow problem equivalent to the L.P. formulation of the matching problem above can be formulated as in Figure 6.

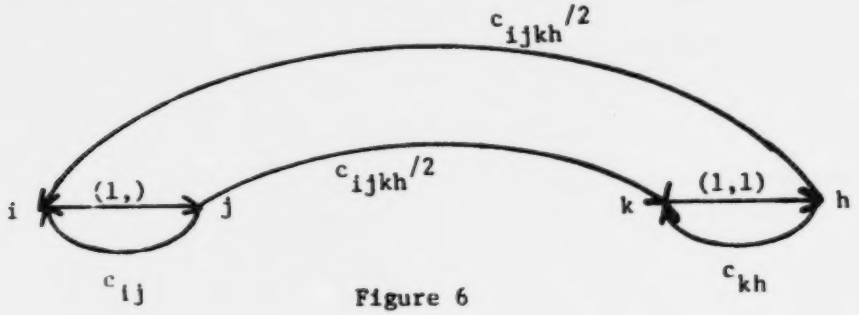


Figure 6

where

- for each existing piece (i,j) nodes i and j and one directed arc (ij) are created with a minimum and maximum flow of one unit, and a cost of 0. An arc j to i is also created to represent the fact that the piece (i,j) may be left as a tripper with a cost c_{ij};
- for each feasible workday composed of two pieces (i,j) and (k,h) two arcs are created, one from j to k and one from k to i, both with a cost $c_{ijkh}/2$.

An equivalent assignment problem to the continuous matching problem can also be generated. The solution of the flow problem corresponds to the solution of the L.P. formulation of the matching problem. Every cycle containing one or two pieces corresponds to either a tripper, a continuous workday or a regular workday. For every cycle containing an even number of pieces, it can be proved that an equivalent solution can be obtained with two pieces per cycle. Finally a cycle containing an odd number of pieces corresponds to a non-integer solution to the L.P. matching problem. We found however that, most of the time, the solution had no odd cycles. When an odd cycle was present, the introduction of a single cut (constraint) in the flow algorithm was enough to get rid of the non-integrality of the solution.

A heuristic takes care of the very few occurences where a single cut does not generate an integer solution for the matching problem. The result that the continuous matching problem is nearly always integer is explained by the fact that the problem is very close to a bipartite matching; pieces of work for the afternoon are matched with either a morning or evening piece. However proceeding with an a priori defined bipartite matching assignment problem rather than with the matching defined previously does not yield as good a solution. Most algorithms (RUCUS, Hamburg, etc.) do however use a bipartite matching approach.

The solution obtained uses only feasible workdays; however constraints (ii) of (5) may not all be respected and several trippers may remain. The heuristic described in the following section is designed to further eliminate the trippers.

A MARGINAL IMPROVEMENT HEURISTIC-The main process of this heuristic involves marginally replacing each partition of each block by an alternate partition. This is achieved as follows:

Step 1 : For each block generate the set B_ℓ of all (if not too many) partitions that use only pieces (i,j) corresponding to a positive x_{ijkh} or x_{khij} in the optimal solution of HASTUS-macro (2) . If insufficient partitions are generated pieces corresponding to null x_{ijkh} with a small reduced cost may be used.

Step 2 : For each block ℓ=1,..,L

a) take out first the partition p_o of block ℓ used in the matching problem

b) consider the resulting set of trippers R_ℓ (composed of trippers in the preceding matching solution and pieces that were matched to pieces of the partition p_o used for block ℓ) ;

c) choose the partition p_k of B_ℓ, that matched with the trippers of set R_ℓ produces the least cost solution (trippers being highly penalized) which improves feasibility if violated. Replace p_o by p_k and update the matching solution accordingly (p_k may equal p_o).

Step 3 : If the solution has been improved (cost is reduced or feasibility improved) after considering each block in turn, go back to Step 2. If not, solve the matching problem (8) and stop.

If after these steps a satisfactory solution is not obtained, the solution may be perturbed in different ways to try to achieve a better solution by reapplying Steps 2 and 3 of the heuristic. For example, we may arbitrarily increase the number of trippers in the matching solution (by removing a certain number of matches) and reapply the heuristic.

This perturbation applied repeatedly has proved useful to generate solutions with no or few trippers.

At this stage, we could also use other marginal improvements heuristics in the literature.

ALGORITHM MODIFICATIONS FOR WORKDAYS WITH THREE PIECES OF WORK-In some cities, workdays with three pieces are permitted and compose in general about ten percent of all workdays. The adaptation of the general strategy described is however straightforward and heuristic in nature. The adaptation of the general formulation (5) is direct;

variables x_{ijkhmn} are created for such feasible workdays. For the HASTUS-macro formulation, the same comment applies : it is necessary however to limit the number of such variables created, considering only the most probable location in the time table for such workdays. After the partition of the blocks and before the matching problem, it is necessary to pre-match two of the pieces of any three piece workday that emerges from the HASTUS-macro solution. These pre-matched pieces are considered as one piece in the matching problem (8). In the marginal improvement heuristic, it is possible to generate additional three piece workdays to reduce the number of trippers ; such a routine exists in the HASTUS program.

RESULTS-A large interactive system has been designed for the implementation of the HASTUS system by the Montréal Transit Commission (C.T.C.U.M.). It is estimated that HASTUS generates a saving of three percent in driver costs. Table 6 gives a comparison of weekday costs for a Montréal garage using manual and HASTUS solutions.

TABLE 6 - Cost comparison (in dollars per day) for week-day for one garage in Montréal

	Manual	HASTUS
Cost of regular hours worked	19,103.75	19,084.32
Cost of trippers	740.81	739.32
Cost of travel time between pieces	682.51	433.11
Over-time cost	231.49	78.73
Guaranteed hours (pieces) cost	82.07	63.80
Guaranteed hours (workday) cost	72.50	153.26
Paid break in three pieces runs	nil	113.24
Layover cost	783.00	740.95
Bonus	8.70	8.70
Spread rate 1 (spread between 10 and 12 hours)	1,063.79	882.61
Spread rate 2 (spread between 12 and 13 hours)	524.61	232.00
Total	23,303.23	22,522.51
Fringe benefits (50%) for additional operators required (2)	0	69.60
TOTAL	23,303.23	22,592.11
Difference		711.12
Percentage of savings		- 3.147

SCHEDULING VEHICLES AND CREWS CONCURRENTLY : A MATCHING BASED APPROACH [12]

In Ball, Bodin and Dial [8], the crew vehicle scheduling problem is formulated as a partitioning problem on a network. The nodes of the network are the driver-trips (d-trips) which represent the minimal amount of work that must be performed by a single crew. A supersource s and a supersink t are added. The network contains two types of arcs:

I) those that represent transitions by a crew and a vehicle, e.g., a crew driving a vehicle from one location to another;
II) those that represent transitions by a crew only, e.g., a crew taking a relief break.

The algorithm first generates a set of pieces, then performs a piece improvement routine and finally combines pieces into runs. The pieces are generated by partitioning the nodes in a network into paths of restricted length. The path length restriction, T, would be the maximum piece time, e.g., 4½ hours.

The major steps of the algorithm are :

Step 1 : Form the crew/vehicle scheduling network but include *only* Type I arcs. Partition the nodes in this network into a set of paths (pieces) whose lengths are less than or equal to T.

Step 2 : Improve the set of pieces by performing combination/splits of pairs of pieces.

Step 3 : Combine pairs of pieces into runs or crew schedules.

Step 1, the piece generation algorithm, starts by partitioning the nodes in the network into a set of levels. The level of a node i equals the number of arcs in the path from s to i that contains the most arcs (see Figure 7). The algorithm iteratively generates larger and larger pieces. At each iteration, it adds the nodes on the next highest numbered level to the network and attempts to merge these nodes into the "partial pieces" already generated. If a node cannot be merged into a partial piece, it becomes the beginning of a new piece.

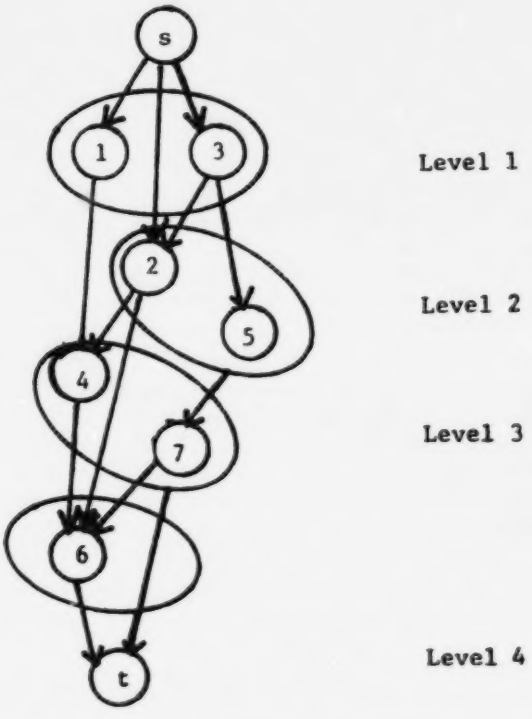
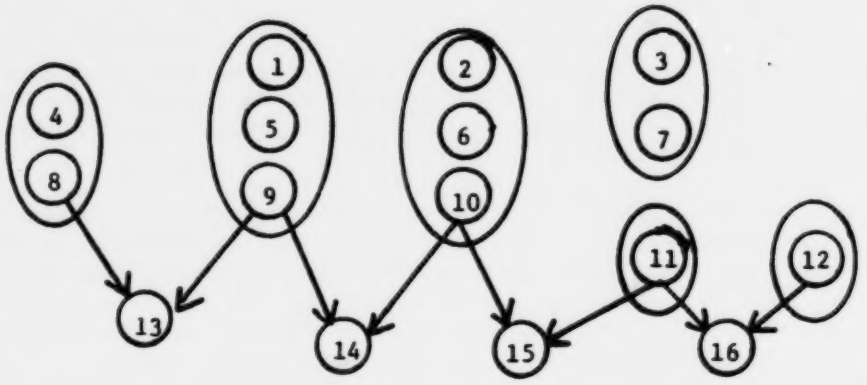


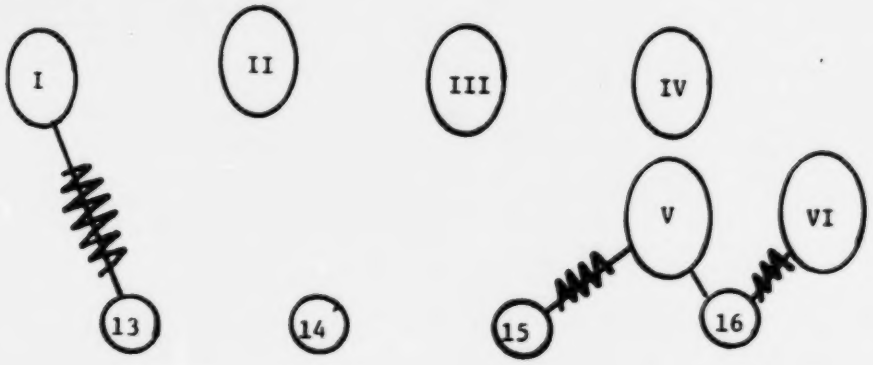
Figure 7 - Partition of Nodes into Levels [12]

The problem of finding the optimal set of merges may be formulated and solved as a matching problem. Figure 8 illustrates this



Partial pieces are circled;
set of node s on level k :
(13, 14, 15 16)

If the max duration T implies no piece can contain more than 3 nodes, then the matching network with optimal solution indicated is:



(indicates arcs in matching)

(continued on next page)

Figure 8 - Matching Problem at Iteration K

New set of pieces :

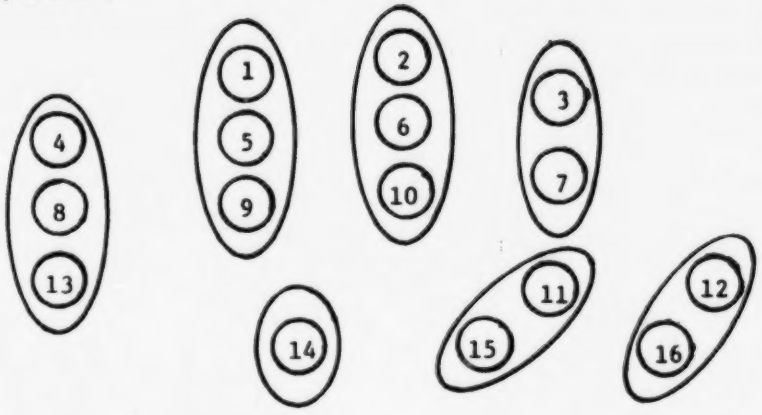


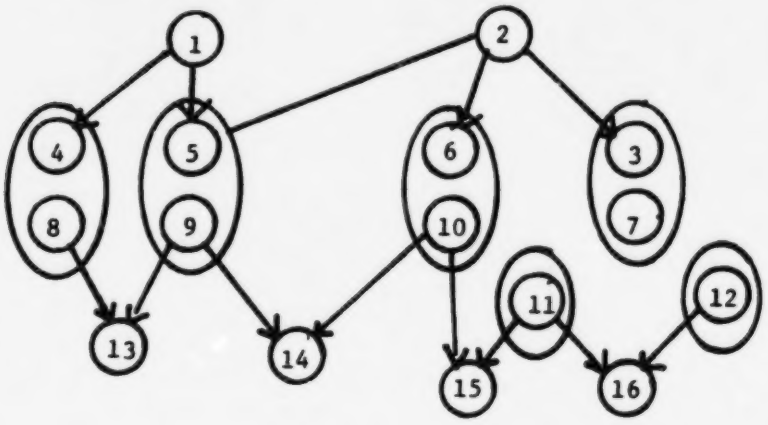
Figure 8 - Matching Problem at Iteration K [12]
(continued)

subproblem for iteration k. The nodes in the matching graph are the partial pieces generated up to iteration k-1 and the nodes on level k. A node corresponding to a partial piece is adjacent to a node corresponding to a node on level k if there exists an arc in the original network pointing out of the last node in the partial piece to the node on level k and if, when the node and arc are added to the partial piece, its duration does not exceed T. If we assign costs of 1 to all arcs and nodes in this network, the minimum cost matching will merge nodes on level k with the partial pieces on level 1 through k-1 in a way that minimizes the number of new pieces generated.

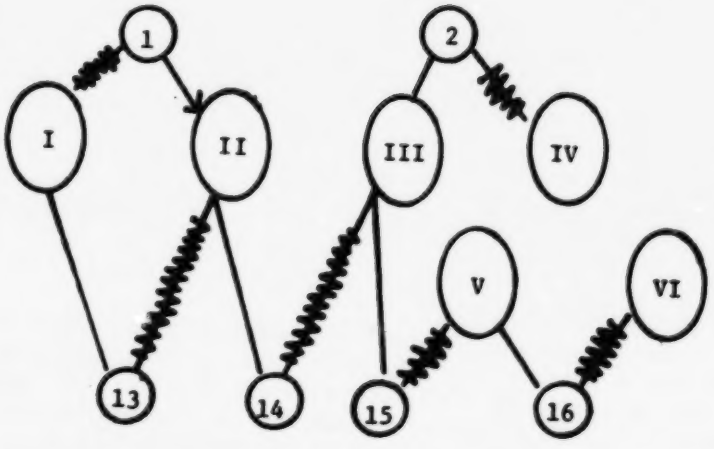
The efficiency of the algorithm is enhanced by allowing the "tops" of pieces to vary when the piece length gets close to T. Figure 9 illustrates this process. When the duration of a piece goes above a lower limit T', (a number less than T but fairly close to T), the first node in the partial piece is expanded. That is, it is no longer fixed as a member of a partial piece, but may become a member of any partial piece to which it is adjacent. At the same time the partial piece is allowed to expand by adding a node to its bottom. No node additions will be allowed unless the new duration of the partial piece is less than T. However, the duration is now measured without including the top node which has been freed. In this manner the algorithm exchanges a new bottom node for a top node, if this process improves the solution. As Figure 9 illustrates, the exchange may be performed using the matching algorithm. (This algorithm can also be used to partition the nodes in an acyclic network into paths restricted in length).

Step 2 initially considers combining and then possibly resplitting every pair of pieces. A value is associated with this combining/splitting operation for each pair of pieces. This value includes components that minimize the number of pieces and minimize the number of short pieces. The problem of determining the best set of combinations/splits can be solved as a matching problem. The nodes are the pieces and the arc costs are the costs of combining/splitting the corresponding pairs of nodes.

Using network from Figure 8 when tops of pieces whose length is greater than 3 nodes are expanded the new network is :



The matching network with optimal solution is :



(continued on next page)

Figure 9 - Matching Problem with Expansion of Tops and Pieces

New set of pieces :

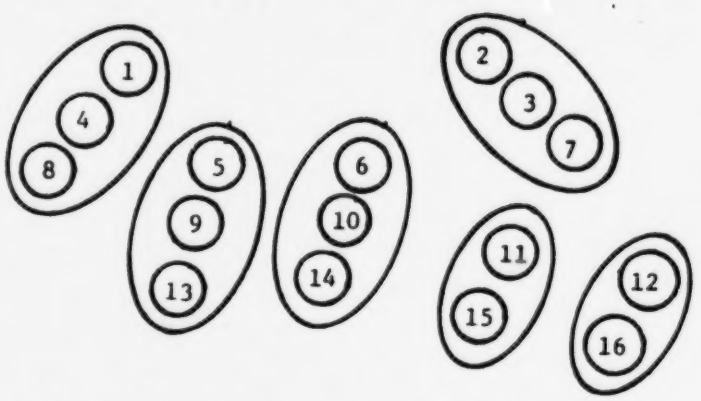


Figure 9 - Matching Problem with Expansion of Tops and Pieces [12] (continued)

As was mentioned earlier, the problem of combining pieces into runs (Step 3) can also be formulated and solved as a matching problem (8). This matching problem may also have side constraints associated with it such as a constraint that at least 50% of all runs must be straight runs.

Although this procedure has not been used in an operational environment, it has been tested on a data base from the City of Baltimore and has produced significant cost savings (see Ball, Bodin and Dial [8]).

THE ROSTERING PROBLEM

A simple rostering problem occurs when a scheduler has to assign days off according to certain rules. Normally the union requires as many weekends off and as many consecutive days off as possible. For its part, management normally requires a minimum number of workdays between sequences of days off. This problem is considered first and referred to as the days off allocation problem. A more complex rostering problem consists of allocating both workdays and days off in order to equalize the workload among the crew. This problem, called the workdays rostering problem, is common in airline crew scheduling and in most European urban transit crew scheduling problem.

THE ALLOCATION OF DAYS OFF-Mathematically this problem can be formulated as in (1) where the variables correspond to sequences of workdays and days off rather than sequences of working periods and breaks. Each constraint ensures that a sufficient number of crews will be present each day. The solution strategy described in the section on scheduling straight workdays thus applies equally here. Very often, the problem is a weekly cyclic problem and heuristic approaches are used. As many as possible Saturdays and Sundays are allocated, then as many Sundays and Mondays, and so on.

When days off must be consecutive, the problem has a nice structure described by Bartholdi et al.[9,10]. If the problem is to

minimize the number of workers with the added proviso that each worker must be allowed two consecutive days off each week the integer linear program (1) becomes:

$$\text{Min} \sum_{i=1}^{7} x_i$$

$$\text{s.t.} \quad \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} x \geq \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} \qquad (9)$$

where each column represents a possible pattern of days off, x_j represents the number of workers on work pattern j and b_i is the number of workers required on day i.

Consider now the complementary problem which is a sort of negative image of the original

$$\text{Max} \sum_{j=1}^{7} y_j = \bar{s}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} y \leq \begin{bmatrix} s-b_1 \\ s-b_2 \\ s-b_3 \\ s-b_4 \\ s-b_5 \\ s-b_6 \\ s-b_7 \end{bmatrix} \qquad (10)$$

Let s be the size of the workforce. Then $s-b_i$ represents the maximum number of workers who may be idle on day i. We have thus formulated the following problem: given a workforce of size s, maximize the number of workers who may be given two consecutive days off each week. Clearly, if for a given s the solution is $\bar{s} < s$, the workforce is too small to meet daily staffing requirement and still allow everyone time off. On the other hand, if for fixed s we find $\bar{s} > s$ this means that too many workers have been assigned time off. This may be easily corrected by simply not assigning workers to days off; $\bar{s} \geq s$ also means that the workforce is adequate to meet the requirements. In fact, we seek the smallest integer $s = s^*$ for which the corresponding solution $\bar{s} \geq s^*$.

Noting that problem (10) has the familiar structure of a maximum matching problem, one can construct a solution strategy based on a binary search on s* and on the solution of the b-matching problems. (see [9]).

A more general strategy is described in [10] for this type of problem when the matrix is row circular, i.e. on each row that 1's occur consecutively when the first and last entries are considered to be consecutive. The matrix in (9) is both row and column circular.

WORKDAY ROSTERING-This combinatorial problem is generally more complex than the preceding one. For example in air crew rostering certain characteristics of crew members (e.g. visas and qualifications for certain flights, periods of unavailability (vacations, training) already assigned for the previous month, etc.) must be taken into account. Other inputs include of course the schedules, credited flight time and actual flight hours for each rotation, the minimum free time needed before and after each rotation, the number of flight legs and undesirability of the rotation. Constraints require a certain number of rest periods at the base (48 or 72 hours for example) each month together with common sense rules for determining minimum and maximum distances allowed between tasks or free periods. In addition, all rotations and reserve duty periods must be assigned and, if by any chance, a rotation or a reserve is not assigned, it must be as short as possible. Overtime must be reduced to a minimum in the objective function as must the number of crew members flying under minimum guaranteed pay. In some problems, crew members' quality of life is also taken into account in the objective function. An attempt is made to avoid repeating the same rotation in the schedule of any crew member, to distribute the number of take-offs and duty days evenly between crew members, to evenly distribute the flight hours paid to crew members and to evenly distribute the rotations considered undesirable by the crew members. Some packages (Selective Bidding Corporation) even take into account the individual preferences of the crew members. North American companies generally try to give the best rosters to the most senior pilots leaving less desirable rosters to more junior pilots.

Graph of the problem-The problem can be described using a graph (fig. 10) consisting of two types of nodes and arcs [3]. Each node R_i represents a rotation or a reserve duty period and the full-line arcs join the mutually compatible rotations and reserve duty periods (i.e., presenting no unauthorized overlapping, respecting the rules of minimum and maximum distance and the statutory rest periods before and after the rotations). These arcs are not oriented. Each node P_j represents a crew member.

The dotted-line arcs join each crew member with the rotations or reserve duty periods which he cannot perform (i.e., these rotations or reserve duty periods are those occuring during a period of unavailability of the crew member or during requested free days or rotations; they may also indicate that the crew member does not have the qualifications needed to perform the rotation).

Solutions to the problem are sets of chains joining compatible rotation and reserve nodes. Each chain is related to a crew member so that no rotations or reserves of the chain are joined with the corresponding crew member node by a dotted-line incompatibility arc. The

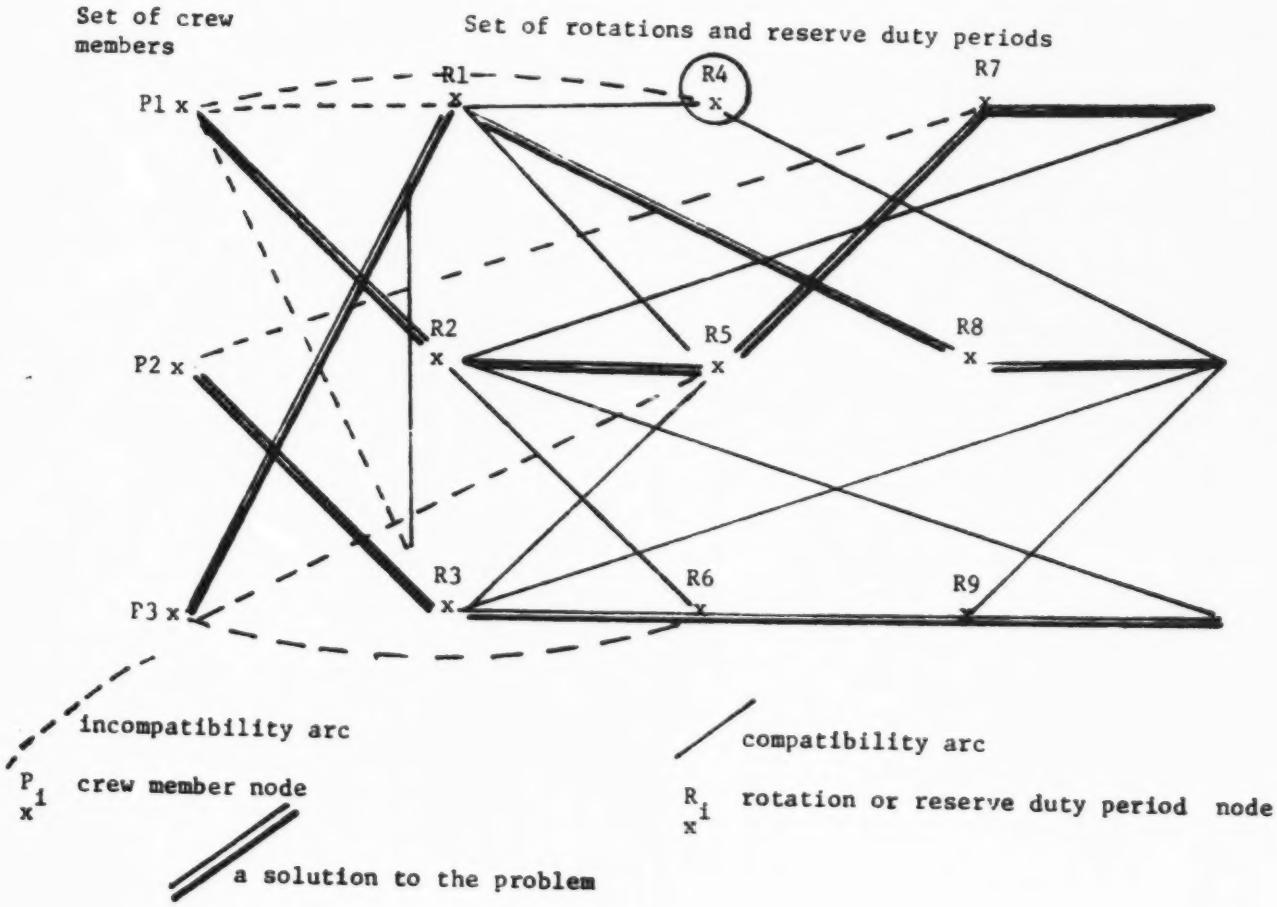


Figure 10 - Graph of the rostering problem [3]

roster consisting of a crew member with his periods of unavailability and a related chain must satisfy the regulations.

The rotations and reserve duty periods not related to a crew member are considered unassigned.

SOLUTION STRATEGIES

In this context, solution strategies are mostly heuristics. Two types of construction heuristic are used. One proceeds from pilot to pilot, forming the complete roster of each pilot before building the roster of another pilot: this is the crew member by crew member method [3]. A more common heuristic however, calls for the daily assignment of the rotations starting on a given day to the rosters in formation: this is the day by day method. Finally, marginal improvement methods can be applied to the solution obtained to get an even better solution. These methods are now described.

Crew member by crew member method [3]-This method consists of the following steps:

Step 1 : A crew member must be isolated from the crew members list. This could be the most senior pilot if he must get the best roster. At Air France [3], in order to get a good solution, the first pilot in the pile is the one who can perform the smallest number of rotations.

Step 2 : The algorithm must assign not-yet-assigned rotations and reserve duty periods to this crew member so that the resulting roster satisfies the constraints and the objectives of the assignment problem. For example, several rosters are generated for that pilot and a selection is made by weighting certain criteria such as
- the total duration should be as high as possible but without overtime
- the assigned rotations should be as long as possible (long rotations are difficult to assign)
- as many as possible rotations and reserve duties must be situated at the peak period of the month.

Step 3 : This pilot is dropped from the list and his roster considered fixed. Return to Step 1 to repeat the process with another pilot.

The difficulty of this approach is that each time a crew member is processed there are fewer and fewer rotations left and it becomes more difficult to form a feasible roster. Sophisticated heuristics must be incorporated in order to ensure a good solution. It is absolutely necessary with such an approach to use marginal improvement heuristics to correct the deficiencies on the last rosters generated.

Day by day methods [3]- The basic day by day method can be described as follows:

Step 1 : Isolate those rotations and reserve duty periods beginning on the first day of the month.

Step 2 : Assign these rotations and reserve periods to the crew members so that the resulting rosters from the tasks already assigned are feasible and satisfy as much as possible the objectives.

Step 3 : Isolate the rotations and reserve duty periods beginning on the next day and go back to Step 2.

In this approach the assignment methods, and the treatment of the problems occurring for the successive days must be clarified. It is in fact important to have a global view of the problem for the month especially when the month includes a peak activity period, or very long rotations. Before considering the assignment method, we must clarify the objective function used in each day by day assignment.

Penalty functions for the day by day assignment problem-The penalty function is necessarily heuristic in nature and depends heavily on the problem under consideration. Nicolletti [40,41] proposes a monthly objective function with penalties calculated as explained by figure 11.

On the horizontal axis, the days of rostering are considered, while the vertical axis measures the quota of a certain factor to be fulfilled by a certain crewman, on the day in the abscissa. The straight line O-T represents the ideal loading along the month of that factor for the crewman considered. For the assignment of activities, during the second day, for example, the cost of a pairing between the crewman considered and an activity having a quantity A of a certain factor, is given by the difference from the diagonal that the crewman had in advance (segment B), the present difference (segment C), and all future differences, which derive as a consequence of the choice considered. All the differences are considered squared in order to weight larger differences more heavily.

This way of calculating the penalty is very convenient and can even be different for each pilot if pilot preferences are to be taken into account. Different weights have to be given to each characteristic for which a penalty is calculated and additional penalties may be added to take into account the desired length of time between restdays at the base.

The assignment algorithm-A regular flow or assignment algorithm can obviously be used at this step. Some heuristics are also proposed in the literature, for example:

- choose the pilot with the highest penalty and assign him to the rotation, reserve duty or sequence of restdays that will reduce the penalty to a minimum,
- once a heuristic assignment is performed, try to exchange assignments in order to improve the solution.

However flow and assignment algorithms are so fast that there is no reason why they should not be used.

Overlapping daily assignment subproblem-In the assignment for day i, Tingley at Swissair [54] proposes the use of all rotations overlapping day i instead of just those starting on day i. The resulting assignment problem is much larger and the solution bound to be better. Only the rotations assigned ending on day i are considered fixed for the next steps. Another improvement, called "preclusion", is also suggested. This technique looks ahead in the tree of future solutions to discover whether a given rotation that starts maybe only on day i+1 or i+2 must be assigned to a given pilot (e.g. whether there is only one possibility for the assignment of this rotation). The computing effort devoted to preclusion should be balanced with the importance of the infeasibilities occurring if no preclusion technique is used.

Marginal improvement methods-All kinds of exchange algorithms well-known in routing and scheduling could be used in this context. An interesting approach however has been applied by Carraresi and Gallo

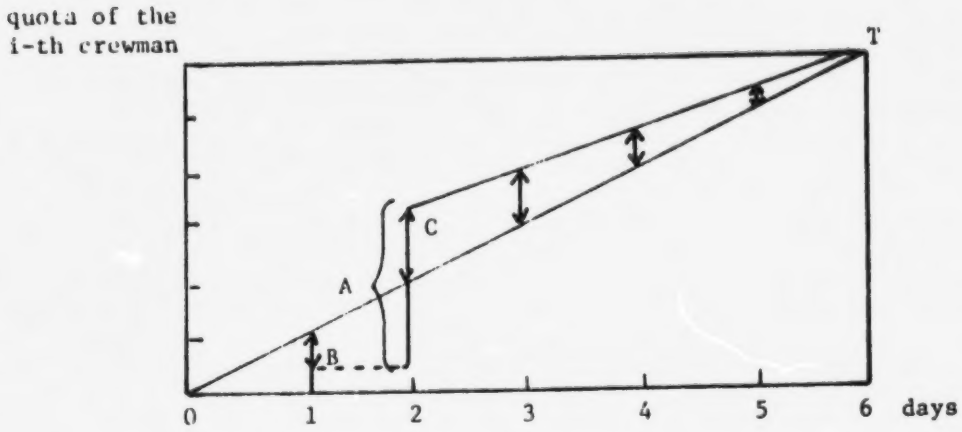


Figure 11 - The basis for the construction of the set of costs [41]

[14] for the transit crew rostering problem in which the tasks are one day duties and the objective is to minimize the maximum total driving time in the period considered. This problem is described as the multi-level bottleneck problem and it is shown that the problem is NP-complete and an asymptotically optimal algorithm is presented for this case. In general the objective function is different but their type of algorithm could be easily adapted to the general situation when starting from a feasible solution.

Sequentially consider each day ℓ in the rostering period.
Option 1 : Take out from each roster the rotation covering (or starting) day ℓ. Define and solve an assignment between the rotations just removed and the partial rosters with the penalty function in use, allowing only feasible assignments.
Option 2 : Define and solve an assignment problem with on one side the partial rosters composed of all the rotations starting before day ℓ and on the other side the partial rosters composed of all the rotations starting on or after day ℓ. Of course only feasible assignments are allowed and the same cost function as in the construction heuristic is used.

If no improvement in the objective function is obtained after all days in the rostering period have been considered the process stops. If an improvement has been achieved, another iteration is performed on the period. From the results of Carraresi and Gallo on their specific problem we can expect this algorithm to perform very well on all rostering types of problems even if the theoretical properties demonstrated by the authors may not hold in all situations.

COMMENTARY

This paper has considered several techniques available for the scheduling of crews. It can be seen that for most problems, a solution approach can be adapted. The real challenge however is probably not to find a solution technique for a given crew scheduling problem. It is much more to find ways to implement the technique so that it is accepted

in an operational environment. Schedulers are very proud of their unique skill in the performance of their job. Any crew scheduling system that threatens their status may be hard to implement in the organization and gain general acceptance. But this is the subject of a lecture in itself!

REFERENCES

1. J. Agard, "Monthly Assignment of Stewards by Computer." 1970 AGIFORS Symposium Proceedings, Terrigal, Australia.

2. N. Amin, J. Ginden, "Computerized Maintenance Staffing." 1976 AGIFORS Symposium Proceedings, Miami, Florida, U.S.A. pp. 505-529.

3. J.L. Antozik, "Automatic Monthly Crew Assignment - A New Approach." 1978 AGIFORS Symposium Proceedings, Vancouver, B.C., Canada, pp. 369-402.

4. E. Baker, "Efficient Heuristic Solutions for the Airline Crew Scheduling Problem." D.B.A. Thesis, University of Maryland, College Park, Maryland, U.S.A., 1979.

5. E. Baker, L. Bodin and M. Fisher, "The Development and Implementation of a Heuristic Set Covering Based System for Air Crew Scheduling." Working Paper #80-015, University of Maryland, U.S.A. December 1980.

6. E. Baker, L. Bodin, W. Finnegan and R. Ponder' "Efficient Heuristic Solutions to an Airline Crew Scheduling Problem." AIIE Transactions, Vol. 11, No. 2, pp. 79-85, 1979.

7. E. Balas and A. Ho, "Set Covering Algorithms Using Cutting Planes, Heuristics and Subgradient Optimization : A Computational Study." Mathematical Programming, Vol. 12, pp. 37-60, 1980.

8. M. Ball, L. Bodin and R. Dial, "A Matching Based Heuristic for Scheduling Mass Transit Crews and Vehicles." Working Paper #81-007, University of Maryland, U.S.A., February 1981.

9. J. Bartholdi III and D. Ratliff, "Unnetworks with Applications to Idle Time Scheduling." Management Science, Vol. 24, pp. 850-858, 1978.

10. J. Bartholdi III, J. Orlin and D. Ratliff, "Cyclic Scheduling Via Integer Programs with Circular Ones." Operations Research, Vol. 28, pp. 1074-1085, 1980.

11. J.-Y. Blais and J.-M. Rousseau, "HASTUS : A Model for the Economic Evaluation of Drivers' Collective Agreements in Transit Companies." INFOR, Vol. 20, pp. 3-15, 1982.

12. L. Bodin, B. Golden, A. Assad and M. Ball, "The State of the Art in the Routing and Scheduling of Vehicles and Crews." Working Paper MS/S #81-035, College of Business and Management, University of Marlyand, U.S.A., 1981. Computer and Operations Research (forthcoming).

13. J. Buhr, "Four Methods for Monthly Crew Assignment - A Comparison of Efficiency." 1978 AGIFORS Symposium Proceedings, Vancouver, B.C., Canada, pp. 403-439.

14. P. Carraresi and G. Gallo, "A Multilevel Bottleneck Assignment Approach to the Bus Drivers Rostering Problem." Publication #180, Centre de recherche sur les transports, Université de Montréal, Canada, 1980.

15. P. Carraresi, G. Gallo and J.-M. Rousseau, "Relaxation Approaches to Large Scale Bus Drivers Scheduling Problems. Transportation Research B, Vol. 16B, pp. 383-397, 1982.

16. B. Couvrey and B. Yansouni, "A Method of Estimating Manpower Levels at an Airport." 1977 AGIFORS Symposium Proceedings, Bad Kissingen, Germany, pp. 340-355.

17. T. Crainic, "Le problème des horaires d'équipages d'une compagnie d'aviation." Publication #122, Centre de recherche sur les transports, Université de Montréal, Canada, 1978.

18. G.B. Dantzig, "A Comment on Edie's 'Traffic Delays at Toll Booths'." Journal of Operations Research Society of America, Vol. 2, 1954.

19. J. Etcheberry, "The Set Covering Problem : A New Implicit Enumeration Algorithm." Operations Research, Vol. 25, pp. 760-772, 1977.

20. A. Gabella, W. Pearce, "Manpower Planning in Telephone Sales Reservations." 1976 AGIFORS Symposium Proceedings, Miami, Florida, U.S.A., pp. 431-472.

21. R. Gerbracht, "A New Algorithm for Very Large Crew Pairing Problems." 1978 AGIFORS Symposium Proceedings, Vancouver, B.C., Canada, pp. 315-342.

22. D. Gretz, "Monthly Crew Planning for Short Haul Rostering." 1973 AGIFORS Symposium Proceedings, Acapulco, Mexico, pp. 150-156.

23. W. Henderson, "Relationships Between the Scheduling of Telephone Operators and Public Transportation Vehicle Drivers." Preprints, Workshop on Automated Techniques for Scheduling of Vehicle Operators for Urban Public Transportation Services, L. Bodin and D. Bergmann, eds., Chicago, Illinois, U.S.A., April 1975.

24. E. Heurgon, "Un problème de recouvrement : l'habillage des horaires d'une ligne d'autobus." Revue Française d'Automatique, d'Informatique et de Recherche opérationnelle, Vol. 6, pp. 13-29, 1972.

25. E. Heurgon, "Preparing Duty Rosters for Bus Routes by Computer." Preprints, Workshop on Automated Techniques for Scheduling of Vehicle Operators for Urban Public Transportation Services, L. Bodin and D. Bergmann, eds., Chicago, Illinois, U.S.A., April 1975.

26. J. Hinson and S. Mulherkar, "Improvements to the Clarke and Wright Algorithm as Applied to an Airline Scheduling Problem." Technical Report, Federal Express Corporation, U.S.A., 1975.

27. J. Hoffstadt, "Computerized Vehicle and Driver Scheduling for the Hamburger Hochbahn Aktiengesellschaft." Computer Scheduling of Public Transport Urban Passenger and Crew Scheduling, A. Wren, ed., North Holland Publishing Co., pp. 35-52, 1981.

28. D. Johnson, "Approximate Algorithms for Combinatorial Problems." Journal of Computer and System Sciences, Vol. 9, pp. 256-278, 1974.

29. A. Land and S. Powell, "Fortran Codes for Mathematical Programming." Wiley, 1973.

30. C. Lemke, H. Salkin and K. Spielberg, "Set Covering by Single Branch Enumeration with Linear Programming Subproblems." Operations Research, Vol. 19, No. 4, pp. 998-1022, 1971.

31. R. Lessard, J.-M. Rousseau and D. Dupuis, "HASTUS I : A Mathematical Programming Approach to the Bus Driver Scheduling Problem." Computer Scheduling of Public Transport Urban Passenger Vehicle and Crew Scheduling, A. Wren, ed., North Holland Publishing Co., pp. 259-268, 1981.

32. S. Lin, "Computer Solutions of the Traveling Salesman Problem." Bell System Technical Journal, Vol. 44, pp. 2245-2269, 1965.

33. F. Marchettini, "Automatic Monthly Cabin Crew Rostering Procedure." 1980 AGIFORS Symposium, New Delhi, India, pp. 23-60.

34. R. Marsten, "An Algorithm for Large Set Partitioning Problems." Management Science, Vol. 20, pp. 774-787, 1974.

35. R. Marsten and F. Shepardson, "Exact Solution of Crew Scheduling Problems Using the Set Partitioning Model : Recent Successful Applications." Networks, Vol. 11, No. 2, pp. 167-177. 1981.

36. M. Mayer, "Monthly Computerized Crew Assignment." 1980 AGIFORS Symposium Proceedings, New Delhi, India, pp. 93-123.

37. R. Moore, J. Evans and H. Ngo, "Computerized Tailored Blocking." 1978 AGIFORS Symposium Proceedings, Vancouver, B.C., Canada, pp. 343-368,

38. J.C. Muller. "Monthly Crew Assignment of Cabin Attendants." 1976 AGIFORS Symposium, Miami, Florida, U.S.A., pp. 613-623.

39. K. Mullin. "An Interactive Approach to Ground Staff Rostering." 1978 AGIFORS Symposium Proceedings, Pugnochiuso, Italy, pp. 383-418.

40. B. Nicoletti, "Automatic Crew Rostering." 1973 AGIFORS Symposium Proceedings, Acapulco, Mexico, pp. 157-182.

41. B. Nicoletti, "Automatic Crew Rostering." Transportation Science, Vol. 9, pp. 33-42, February 1975.

42. B. Nicoletti, L. Natali and M. Scalas, "Manpower Staffing at an Airport : A Generalized Set Covering Problem." 1976 AGIFORS Symposium Proceedings, Miami, Florida, U.S.A., pp. 473-503.

43. M. Parker and B. Smith, "Two Approaches to Computer Crew Scheduling." Computer Scheduling of Public Transport Urban Passenger Vehicle and Crew Scheduling, A. Wren, ed., North Holland Publishing Co., pp. 193-222, 1981.

44. R.A.T.P., "Conditions de travail des machinistes européens". Enquête 1978, RP-2, Paris, 1979.

45. J. Roy and M. Ravacley, "A Customer Selectivity Approach to Operations at CN Express." INFOR, Vol.20, No. 1, pp. 28-39, 1982.

46. J. Rubin, "A Technique for the Solution of Massive Set Covering Problems with Application to Airline Crew Scheduling." Transportation Science, Vol. 7, pp. 34-48, 1973.

47. J. Rubin, "A Survey of Techniques for Airline Crew Scheduling." Working Paper, Cambridge Scientific Center, Cambridge, Mass., U.S.A., 1975.

48. D. Ryan and B. Foster, "An Integer Programming Approach to Scheduling." Computer Scheduling of Public Transport Urban Passenger Vehicle and Crew Scheduling, A. Wren, ed., North Holland Publishing Co., pp. 269-280, 1981.

49. H. Salkin, "Integer Programming." Addison-Wesley, 1975.

50. M. Segal, "The Operator-Scheduling Problem : A Network-Flow Approach." Operations Research, Vol. 22, pp. 808-823, 1974.

51.G.P. Sharp, "Constraints for Scheduling Operators for Urban Public Transit Systems". Preprints of the Workshop on Automated Techniques for Scheduling of Vehicle Operators for Urban Public Transportation Services, Chicago, Illinois, U.S.A., April 1975.

52. W. Shepardson and R. Marsten, "A Lagrangian Relaxation Algorithm for the Two Duty Period Scheduling Problem." Management Science, Vol. 26, pp. 274-281, 1980.

53. G.A. Tingley, "The Swissair System of Assigning Rotations to Pilots by Computer." 1975 AGIFORS Symposium Proceedings, Rotorua, New Zealand, pp. 475-512.

54. G. Tingley, "Still Another Solution Method for the Monthly Aircrew Assignment Problem". 1979 AGIFORS Symposium Proceedings, Pugnochiuso, Italy, pp. 145-203.

55. R. Ward, P. Durant and A. Hallman, "A Problem Decomposition Approach to Scheduling the Drivers and Crews of Mass Transit Systems." Computer Scheduling of Public Transport Urban Passenger Vehicle and Crew Scheduling, A. Wren, ed., North Holland Publishing Co., pp. 297-312, 1981.

56.E. Wilhelm, "Overview of the Rucus Package Driver Run Cutting Program." Preprints, Workshop on Automated Techniques for Scheduling of Vehicle Operators for Urban Public Transportation Services, L. Bodin and D. Bergmann, eds., Chicago, Illinois, U.S.A., 1975.

57. S. Young II, "Shift Scheduling of Nonflight Personnel : Two Methods." 1979 AGIFORS Symposium Proceedings, Pugnochiuso, Italy, pp. 419-446.